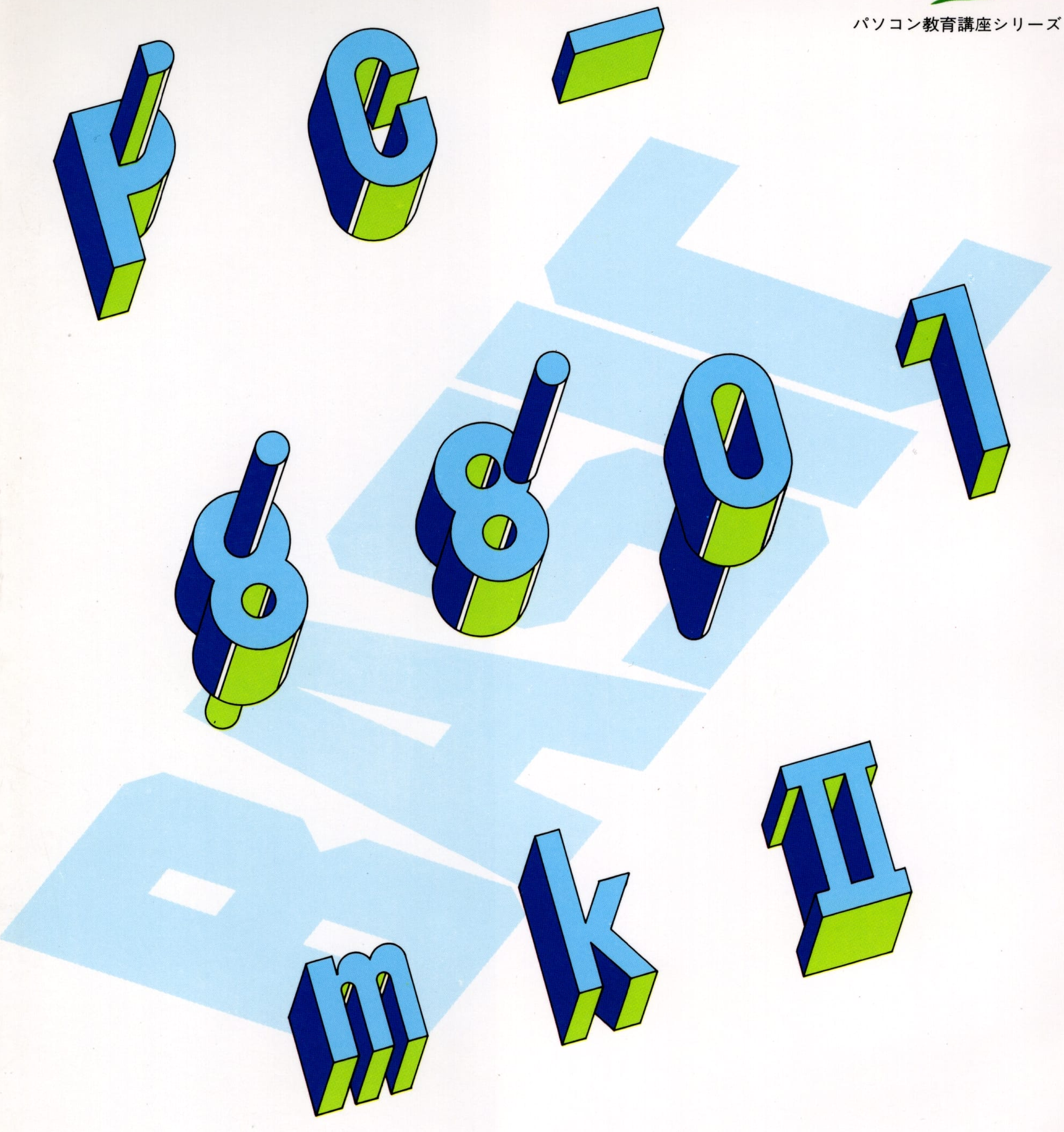
パソコン教育講座シリーズ



PC-8801mkⅡ（マーク）の使い方 下巻

工学博士 松尾三郎 監修

ランダムファイル、グラフィック、割り込み機能についてのプログラムの作り方や考え方と、
ディスクコマンドやユーティリティの使い方について、身近な例を用いて設問形式で詳しく解説しています。

EDC 電子開発学園

パソコン教育講座シリーズ
PC-8801mkⅡ（マーク）の使い方 下巻
EDC 電子開発学園

電子開発学園出版案内〔全巻B5判〕

パソコン教育講座シリーズ

NEC PC-8000 Series

BASIC入門編	(定価 1,700円)
BASIC初級編	(定価 1,900円)
フロッピーディスクプログラミング編	(定価 2,300円)

PC-8001mkII BASIC上巻	(定価 2,300円)
PC-8001mkII BASIC下巻	(定価 2,500円)

PC-8800 BASIC入門編	(定価 1,700円)
PC-8800 BASIC初級編	(定価 1,900円)
PC-8800 BASIC中級編	(定価 2,300円)

PC-8801mkIIの使い方 上巻	(定価 2,300円)
PC-8801mkIIの使い方 下巻	(定価 2,500円)

N5200/05 LANシリーズの使い方	(定価 1,900円)

PC-8801mk II(マーク)の使い方 下巻

工学博士　松尾三郎　監 修

電子開発学園

監修者のことば

ここ数年，パソコン（パーソナル・コンピュータ）の普及には目を見はるものがあります。マイコン（マイクロ・コンピュータ）とかパソコンとか呼び名も不統一なまま世に出たパソコンもあっという間に，小学生はおろか幼稚園の子供からお年寄りまでが，これを操作する時代になりました。

技術革新によって精度の高い超LSIなどの半導体技術が高度に進んだ結果，手のひらに乗るほどのコンピュータが実現し，値段も安くなって一般大衆の商品となりました。

このようにパソコンのハードウェア技術は急速に進歩しましたが，半面，これを使いこなす技術であるソフトウェアの開発はあくまで人の頭脳に頼るところから，需要に十分応えられないのが現状です。コンピュータを生かすも殺すもソフトウェアの成否にかかっています。いくら高性能のパソコンでもソフトウェアがなければ，ただの金属のかたまりに過ぎません。ソフトウェア技術者の能力が高度に発揮されて初めて良質のソフトウェアが作り出されるのです。

私はハードウェア，ソフトウェアという言葉自体が普及していなかった15年前，今日のこのコンピュータ時代の到来を予見し，将来の日本の情報化の一助とするためソフトウェア技術者を養成する電子開発学園を創設しました。全国で８校のコンピュータ専門の専修学校を開き，これまでに２万余人の技術者を世に送り出しました。同時にまた，ソフトウェア技術者を養成するには実務教育が必要であるため，日本電子開発株式会社ならびにソフトウェア・コンサルタント株式会社を設けて教育と実務を直結させる産学協同の実践を図っております。

パソコン全盛の時を迎えソフトウェア技術の啓蒙普及の必要性をますます痛感しています。そこで，これまで蓄積してきた教育現場での経験や技術開発の実績に基づいて，自宅学習でもご理解いただけるような「パソコン教育講座シリーズ」のテキストを刊行しました。「PC-8801mkIIの使い方」はこのシリーズの一環をなすものです。

本書は，発売以来大変ご好評をいただいておりますPC-8800 BASIC入門編，初級編，中級編を全面的に改版の上，より使いやすさをモットーにPC-8801mkII中級用テキストとして更に内容を充実させたものにしました。必ずや皆様のお役に立つものと信じております。

このテキストをもとに，一日も早くパソコンを自分のものにし，自由自在に使いこなすようになっていただければ，あなたの人生に新しい世界が開かれることは間違いありません。

1984年12月

松尾三郎

はじめに

最近では，OA化ブームを映してだれでも一度はパソコンに目を向けるようになり，必ず目に(または耳に)するのがBASICです。このBASICは，もともと初心者用の言語として作られたものであり，コンピュータで使われているいろいろな言語の中では最も初心者向きであるといえます。

そのために，BASICは機能性に乏しいと誤解される場合がありますが，実際には命令や機能が多岐にわたり，BASICを十分理解した上で利用すると，使い方によっては非常に高度な処理も可能になります。

しかし，初心者にとっては一つ一つの命令がどんな働きをするのか，または，どんなときにどの命令を使えばよいのか覚えるのはたいへん難しいことです。ましてや，プログラムの作り方や考え方となるとなおさらといえましょう。

そこで，本書は身近なテーマを使って命令の働きやその書き方からプログラムの作り方までを段階的に取り上げた設問を通して，操作手順や実行したときの結果例を掲げてわかりやすく解説しています。

本書は，1～4章と練習問題，Q&Aおよび付録で構成されており，その主な内容は次のとおりです。

1章はランダムファイル，2章はPC-8801mkIIのグラフィックス機能，3章はディスクコマンドとユーティリティプログラムの操作方法，4章はPC-8801mkIIが持つ割り込み処理についてそれぞれ説明しています。

なお，各章の執筆は電子開発学園出版局の豊島俊郎が担当し，全体のとりまとめは同局の山縣恵昭が行いました。

さらに，このテキストは基本的にはPC-8801mkII model30（5インチ・ミニ・フロッピーディスク装置2ドライブを本体に内蔵）のマシンを前提に説明されていますが，プログラムを確認するために使用したパソコンの構成は次のとおりです。

①本体――――――――――――PC-8801mkII model30

②CRTディスプレイ――――――PC-8853（高解像度のカラーディスプレイ）

③プリンタ――――――――――PC-8822（漢字ドットプリンタ）

本書を刊行するにあたり，ソフトウェアコンサルタント株式会社，株式会社イーディシー，日本電子開発株式会社各社の技術部には多大の応援をいただきました。厚くお礼申し上げます。

目　　次

第1章　ランダムファイル

上巻では，フロッピーディスクにデータを記録する方法としてシーケンシャルファイルを学習しました。ところが，シーケンシャルファイルは，発生したデータを順番にフロッピーディスクに出力してできるファイルであるため，データを出力したときの順序でしか読むことができないという制約がありました。

そこで，ここではファイルから自由にデータを読み書きすることができ，かつフロッピーディスクの最大の特長ともいえるランダムファイル(乱編成ファイルとも呼ばれ，中型以上のコンピュータで利用されている相対編成ファイルとよく似ています)について学習することにします。

1.1　ランダムファイルとは

ランダムファイルは，発生したデータをファイルの中のどの位置へでも自由に書いたり読んだりすることができます。これは，ランダムファイルを構成するレコードの桁数が「256桁」と決められているためです。つまり，一つのレコードの桁数が256桁に固定されている（「固定長レコード」といいます）ために，何番目のレコードの読み書きを行うかを指定することによってそのレコードのファイル中での位置がわかるからです。たとえば，5番目のレコードといった場合は，ファイルの先頭から「256＊(5−1)＋1(＝1025)」桁目から256桁のデータになります。

シーケンシャルファイルのデータ記録

1件目	2　件　目	3件目	4 件 目	…………………………

・おのおののレコードの桁数は，内容によって変わる。

ランダムファイルのデータ記録

1　件　目	2　件　目	3　件　目	4　件　目	…………………………

・レコードの桁数が決まっているので，何番目であるかを指定することによって自由に読み書きすることができる。
・記録できるデータの件数を前もって計算することができる。

上巻で学習したシーケンシャルファイルは，フロッピーディスクにデータを記録するときのレコードの長さがデータの内容によっておのおの異なりました。(「可変長レコード」といいます)

したがって，一つのファイルに何件のデータが入るかを前もって計算しておくことはできません。

ところが，ランダムファイルでは一つのレコードの長さが決まっているので，何件のデータを記録することができるかを計算することができます。ただし，ファイルに入るレコードの数はフロッピーディスクの空き領域がどれだけあるかによって決まるので，あらかじめフロッピーディスクの空き領域を「DSKF」関数で調べておく必要があります。

(例)　<u>ドライブ2に入っているフロッピーディスクの空き領域を調べる場合</u>

? DSKF(2)　　または　　A=DSKF(2)：? A

DSKF関数で求められる値は，「クラスタ」と呼ばれる単位の値になります。1クラスタには「2048バイト(桁)」のデータを記録することができます。(「1.4.1　フロッピーディスクの構造」を参照)

データ記録用のフロッピーディスクとしてフォーマッティングしたばかりのミニフロッピーディスクの空き領域は「158クラスタ」です。

したがって，1枚のミニフロッピーディスクに記録できるランダムファイルのデータ件数は次のようになります。

<u>ミニフロッピーディスクに記録できるデータの件数</u>

158(クラスタ)＊2048(バイト)￥256(バイト)＝1264(件)

- 256(バイト)：1レコードのバイト数
- 158(クラスタ)＊2048(バイト)：1枚のミニフロッピーディスクに記録できる総バイト数

8インチの標準フロッピーディスクに記録できるデータの件数もミニフロッピーディスクの場合と同じようにして計算することができますが，8インチ標準フロッピーディスクでは1クラスタに記録できるデータは「6656バイト(桁)」になります。また，データ記録用のフロッピーディスクとしてフォーマッティングしたばかりの8インチ標準フロッピーディスクの空き領域は「151クラスタ」です。

<u>8インチ標準フロッピーディスクに記録できるデータの件数</u>

151(クラスタ)＊6656(バイト)￥256(バイト)＝3926(件)

※　ここで求めたデータ件数は，フロッピーディスクに一つのランダムファイルを記録したときの最大のデータ件数です。したがって，他のファイルと一緒に記録した場合はもっと少なくなります。

1.2　ランダムファイルに向いている仕事

これまでに述べてきたように，ランダムファイルはシーケンシャルファイルと違って，ファイルの中のどの位置でも自由に読み書きできます。この性質をうまく利用することによって，ランダムファイルの効果的な使い方ができます。

たとえば，得意先の情報(得意先名，住所等)や商品の情報(商品名，単価，在庫数)などを記録しておく台帳のようなものとして利用すると，個別の情報を位置指定(得意先コードや商品コード)によって即座に取り出すことができます。このように台帳として利用するファイルのことを，一般には「マスタファイル」とよびます。ランダムファイルは，このようなマスタファイルの記録ファイルとして最も適しています。

ランダムファイルは，レコードの番号を1から順番に指定していくことによってシーケンシャルファイルのように使うこともできるので，シーケンシャルファイルの代用をすることができますが，1レコードの記録に必ず256桁を使ってしまうので，履歴ファイルなどのように，順番どおりにデータを処理していく場合にはシーケンシャルファイルの方が無駄がなくて便利です。

1.3　プログラム例(1)　――成績管理プログラム

ここでは，ランダムファイルの作成から基本的な利用の仕方までを，テストの成績管理を例にとって説明します。

1.3.1　成績管理プログラムの概要

ここで行う成績管理は，1クラス分(最大50名)の生徒の主要5教科の得点と平均点をランダムファイルに記録しておき，生徒個人の情報の取り出しやクラス全体の一覧表の出力を行うものです。この節の学習で出力する一覧表のサンプル印字と，生徒個人の情報を表示したときの画面イメージを次に示します。

＜成績一覧表＞

中　間　考　査　成　績　一　覧　表

番号	氏　　名	国語	数学	理科	社会	英語	合計	平　均
1	ﾀｶﾊｼ ﾀﾂﾋｺ	87	54	20	54	95	310	62.0
3	ﾀﾅｶ ﾕﾀｶ	60	80	50	50	80	320	64.0
5	ﾊﾗﾀﾞ ｲﾁﾛｳ	90	89	75	54	100	408	81.6
7	ﾔﾏﾀﾞ ﾀﾛｳ	80	91	75	88	95	429	85.8
9	ﾜﾀﾅﾍﾞ ｼﾞﾛｳ	75	75	68	98	58	374	74.8
31	ｶﾜｲ ﾖｼｴ	60	85	95	77	84	401	80.2
33	ｻﾄｳ ｼﾞｭﾝｺ	40	69	45	45	45	244	48.8
35	ｽｽﾞｷ ﾋﾛｺ	20	64	100	65	54	303	60.6
37	ﾏﾂｼﾀ ﾉﾘｺ	64	66	57	84	84	355	71.0
39	ﾓﾘｶﾜ ｱｲｺ	39	45	98	100	54	336	67.2
10人	ク　ラ　ス　平　均	61.5	71.8	68.3	71.5	74.9	348.0	69.6

＜生徒個人の情報表示＞

```
セ イ セ キ デ ー タ ノ ニ ュ ウ リ ョ ク

1. ﾊﾞﾝｺﾞｳ(999-->ｵﾜﾘ) ? 1

2. ｼﾒｲ      ? ﾀｶﾊｼ ﾀﾂﾋｺ

3. ｺｸｺﾞ     ? 87

4. ｽｳｶﾞｸ    ? 54

5. ﾘｶ       ? 20

6. ｼｬｶｲ     ? 54

7. ｴｲｺﾞ     ? 95
----------------------------------------
   ｺﾞｳｹｲ  = 310   ---> ﾍｲｷﾝ =  62.0

ﾆｭｳﾘｮｸﾊ ｺﾚﾃﾞｲｲﾃﾞｽｶ (y/n) ? y
```

ランダムファイルに格納する成績データの種類とそれぞれの項目の順序を次に示します。

＜項目の順序＞

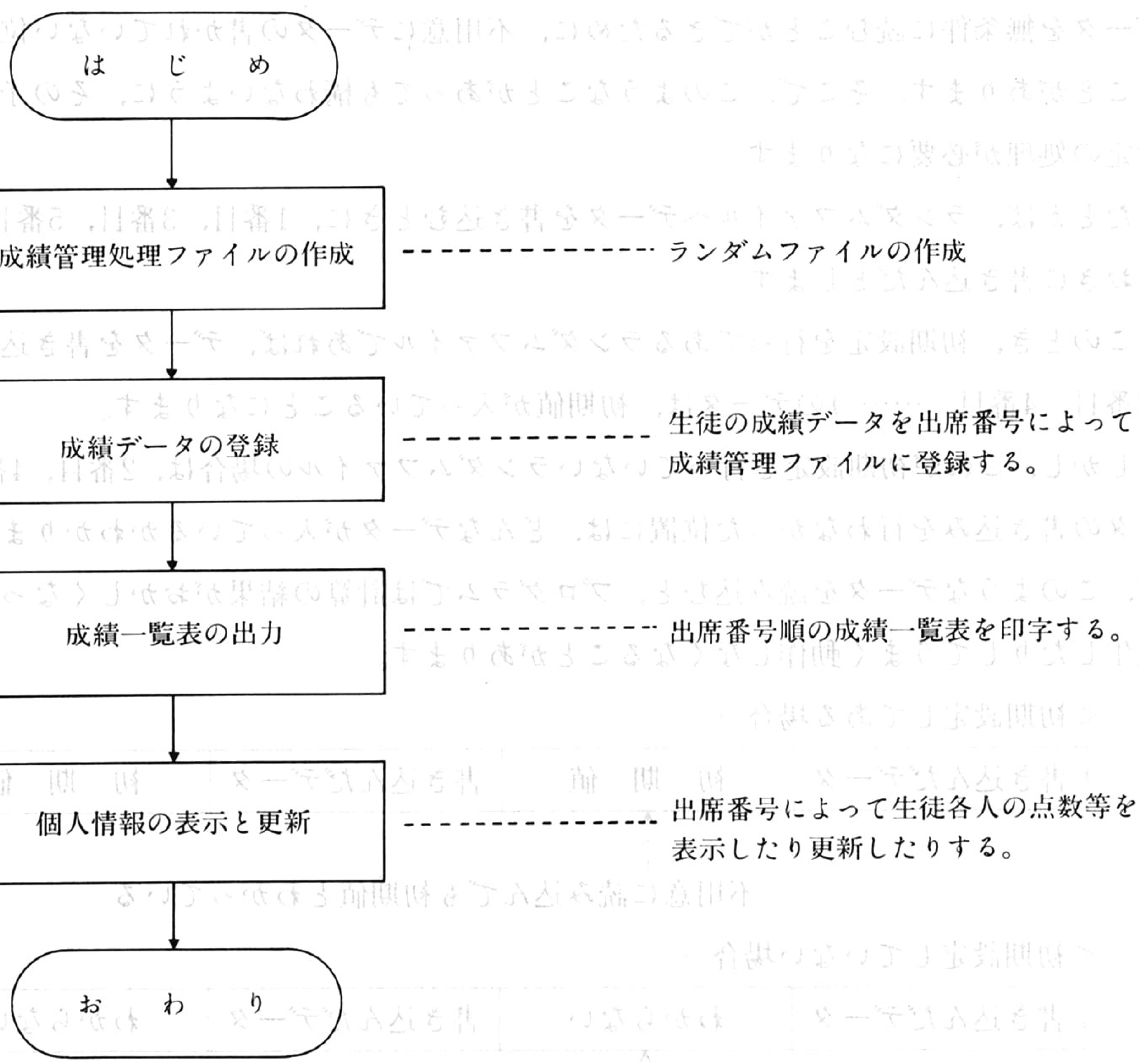
識別コード	氏名	国語	数学	理科	社会	英語	合計	平均
(文字)	(文字)	(数値)	(数値)	(数値)	(数値)	(数値)	(数値)	(数値)

※　項目の中で，識別コードとあるものはそのレコードに正しいデータがセットされているかどうかを判定するためのものです。(1.3.2　参照)

また，「合計」や「平均」は計算で求まるので，一般的にファイルに記録することはしませんが，このテキストでは，後述の「1.3.6　成績データの修正方法」で利用できるようにするため記録することにします。

それでは，先に示した成績一覧表の出力や個人情報の表示が行えるようになるまでのおおまかな手順を次に示します。

＜成績管理プログラムの手順＞



1.3.2 成績管理ファイルを作るには（ランダムファイルの作成）

ランダムファイルを新たに作成するときは，その前に一度だけそのファイルへ登録したいデータの最大件数の分だけを，あらかじめ初期設定(イニシャライズともいいます)しておかなければなりません。

ここでは，ランダムファイルの初期設定のプログラムについて学習することにします。

問題 1　1クラス分(最大50人)の成績データが記録できるような成績管理ファイルを作成しなさい。

【説　明】

N_{88}DISK-BASICでは，ランダムファイルをシーケンシャルファイルと同様に，初期設定なしでも使用することができます。

しかし，ランダムファイルでは一つ一つのレコードの長さが固定されていて指定された位置のデータを無条件に読むことができるために，不用意にデータの書かれていない位置を読んでしまうことがあります。そこで，このようなことがあっても構わないように，その予防策として初期設定の処理が必要になります。

たとえば，ランダムファイルへデータを書き込むときに，1番目，3番目，5番目，…… と，一つおきに書き込んだとします。

このとき，初期設定を行ってあるランダムファイルであれば，データを書き込まなかった位置(2番目，4番目，……)のデータは，初期値が入っていることになります。

しかし，これが初期設定を行っていないランダムファイルの場合は，2番目，4番目，…… のデータの書き込みを行わなかった位置には，どんなデータが入っているかわかりません。したがって，このようなデータを読み込むと，プログラムでは計算の結果がおかしくなったり，エラーが発生したりしてうまく動作しなくなることがあります。

<初期設定してある場合>

書き込んだデータ	初　期　値	書き込んだデータ	初　期　値	………

↑

不用意に読み込んでも初期値とわかっている

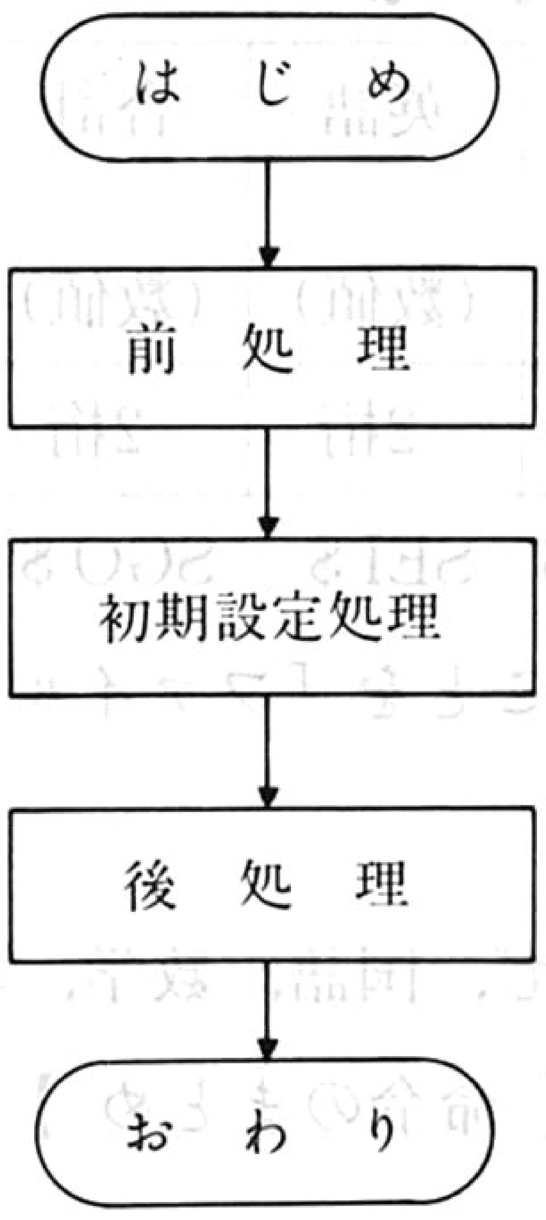
<初期設定していない場合>

書き込んだデータ	わからない	書き込んだデータ	わからない	………

↑

不用意に読み込んだときに値がわからない

それでは，成績管理ファイル(ランダムファイル)を作成するための処理の手順を次に示します。



【前処理】

初期設定処理を行う前の準備作業(通常，プログラムの「初期処理」といいます)として，ファイルのオープンと格納する項目についての定義を行います。

ファイルのオープンを行うには，OPEN命令(【命令のまとめ】を参照)を使います。

項目の定義を行うには，FIELD#という命令(【命令のまとめ】を参照)を使います。

設問 1-1 成績管理ファイルをオープンするために，次の命令をキーインして下さい。

```
10 OPEN "2:SEISEK.MST" AS #1
```

〔キーイン後の説明〕

① この命令は，ドライブ2のフロッピーディスクに「SEISEK.MST」という名前のファイルを，ランダムファイルとしてファイル番号1でオープンします。

② ここでは，成績管理ファイルの名前を「SEISEK.MST」にしています。

設問 1-2 成績管理ファイルの項目の定義を行うために，次の命令をキーインして下さい。

```
20  FIELD#1,1 AS SFG$,20 AS SNM$,2 AS SKO$,2 AS SSU$,2 AS SRI$,2 AS SSHA$,2 AS S
EI$,2 AS SGO$,4 AS SHE$
```

〔キーイン後の説明〕

① ここで定義した成績管理ファイルの項目は，次のようになります。

識別コード（文字）	氏名（文字）	国語（数値）	数学（数値）	理科（数値）	社会（数値）	英語（数値）	合計（数値）	平均（数値）
1桁	20桁	2桁	2桁	2桁	2桁	2桁	2桁	4桁
SFG＄	SNM＄	SKO＄	SSU＄	SRI＄	SSHA＄	SEI＄	SGO＄	SHE＄

※ このようにファイルのなかの項目の形式を表す図のことを「ファイルレイアウト」といいます。

② ランダムファイルでは文字型変数しか許されていないので，国語，数学，理科，社会，英語，合計，平均の各項目も，文字型変数で定義します。(【 命令のまとめ 】FIELD#命令を参照)

③ 数値は， MKI＄，MKS＄，MKD＄という関数(【 命令のまとめ 】を参照)を使って文字型に変換することができます。

④ 文字型に変換された数値は，CVI, CVS, CVDという関数(【 命令のまとめ 】を参照)を使って元の数値型に戻すことができます。

⑤ 国語，数学，理科，社会，英語，合計の各項目は，整数型の値(－32768～32767)で十分なので，文字型の桁数として2桁になります。

⑥ 平均は，小数の値を持つので単精度になります。したがって，文字型の桁数として4桁になります。

⑦ 各項目の変数名は，すべて「S」で始まるようにしています。 これは，プログラムが長くなった場合に，「S」で始まる変数は成績管理ファイルの項目であるということがすぐわかるように，ファイル名「SEISEK.MST」の頭文字「S」に合わせているからです。

【 初期設定処理 】

初期設定処理では，FIELD#命令で定義した各項目に初期値をセットして，50件分(50名分)のデータをファイルへ書き込みます。

FIELD#命令へのデータ(初期値)のセットは，LSETまたはRSETという命令(【 命令のまとめ 】を参照)を使います。

ランダムファイルへのデータの書き込みは，PUT#という命令(【 命令のまとめ 】を参照)を使います。

設問 1-3 各項目に初期値を設定するために，次の命令をキーインして下さい。

```
30 LSET SFG$="0"
40 LSET SNM$=SPACE$(20)
50 LSET SKO$=MKI$(0)
60 LSET SSU$=MKI$(0)
70 LSET SRI$=MKI$(0)
80 LSET SSHA$=MKI$(0)
90 LSET SEI$=MKI$(0)
100 LSET SGO$=MKI$(0)
110 LSET SHE$=MKS$(0)
```

〔キーイン後の説明〕

① FIELD#命令で定義した各項目の初期値をセットする命令として，ここではLSET命令を使っています。

② LSET命令は，項目へのデータセットを左詰め(LEFT)で行うものです。逆に，RSET命令は，項目へのデータセットを右詰め(RIGHT)で行うものです。

一般に，RSET命令は項目のセットに特別な意図を持たせるときにだけ使います。

③ 初期値として，識別コードには文字の「0」，氏名には20桁の空白(ここでは，1桁でもよい)，国語，数学，理科，社会，英語，合計，平均の各項目には数値の「0」をセットしています。

設問 1-4 50件分のデータの初期設定を行うために，次の命令をキーインして下さい。

```
120 FOR I=1 TO 50
130   PUT#1,I
140 NEXT I
```

〔キーイン後の説明〕

① 50件分のデータの書き込みを行うものです。

② 変数「I」は，ファイル中でのデータ記録位置を指定します。

③ PUT#命令でファイルに書き込まれるデータは，FIELD#命令で定義された各項目にセットされている値です。これらの値は，PUT#命令で書き込みを行った後でも変わりません。

※ PUT#命令の前に空白を入れて他の命令と記述位置をずらしていますが，これはプログラムリストをみるときに，行番号120～140のFOR～NEXT命令の中の処理であることをわかりやすくするためです。

このテキストでは，プログラムの見やすさや処理イメージをつかみやすくするため，プログラムのリストにはできるだけこのような工夫をしています。

【 後処理 】

初期設定処理を行った後の処理(通常，プログラムの「終了処理」といいます)として，ファイルのクローズを行い，END命令を実行します。

ランダムファイルのクローズも，シーケンシャルファイルと同様CLOSE命令で行います。

設問 1-5 終了処理を行うために，次の命令をキーインして下さい。

```
150 CLOSE
160 END
```

〔キーイン後の説明〕

① ランダムファイルをクローズするためのCLOSE命令の形式や使い方は，シーケンシャルファイルと同じです。

【 プログラムの確認 】

設問 1-6 正しくキーインできているかどうかを確認するために，キーインしたプログラムのリストを表示して下さい。

<確認用リスト>

```
10 OPEN "2:SEISEK.MST" AS #1
20  FIELD#1,1 AS SFG$,20 AS SNM$,2 AS SKO$,2 AS SSU$,2 AS SRI$,2 AS SSHA$,2 AS S
EI$,2 AS SGO$,4 AS SHE$
30 LSET SFG$="0"
40 LSET SNM$=SPACE$(20)
50 LSET SKO$=MKI$(0)
60 LSET SSU$=MKI$(0)
70 LSET SRI$=MKI$(0)
80 LSET SSHA$=MKI$(0)
90 LSET SEI$=MKI$(0)
100 LSET SGO$=MKI$(0)
110 LSET SHE$=MKS$(0)
120 FOR I=1 TO 50
130   PUT#1,I
140 NEXT I
150 CLOSE
160 END
```

【 プログラムの実行 】

設問 1-7 正しくキーインできていたら，プログラムを実行して下さい。

〔実行後の説明〕

① 「ドライブ2」のフロッピーディスクに「SEISEK.MST」という名前で成績データを登録するランダムファイルが次の形式で作られます。

識別コード	氏　名	国語	数学	理科	社会	英語	合計	平均	未使用	識別コード
”０”	空白	0	0	0	0	0	0	0		”０”
(1桁)	(20桁)	(2桁)	(2桁)	(2桁)	(2桁)	(2桁)	(2桁)	(4桁)	(219桁)	(1桁)

1件目のデータ　　2件目のデータ

設問 1-8 正しく実行できたら，プログラムをドライブ 2 のフロッピーディスクへ「MONDAI.1」という名前で保存して下さい。

<実行結果例>

```
save "2:MONDAI.1"
Ok
```

【補足説明】

ランダムファイルにデータを書き込むときには，指定したレコード番号の位置に既にデータが登録されているかどうかを調べる必要があります。もしそれを調べないでデータを書き込むと，その位置に登録されていたデータは消えてしまい，後から書かれた内容に置き替わってしまいます。これでは，どんなデータも安心して登録しておくことはできません。

そこで，このような場合，一般に「識別コード」とか「区別コード」とか「フラッグ」などと呼ばれる（ここでは「識別コード」と呼んでいます）データを区別する１桁のコードを一つ一つのデータに必ず付けておきます。そして，データを書き込むときには必ずこの識別コードを調べたうえで行うようにすれば，既に登録されているデータを保護することができます。

【 命令のまとめ 】

ここでは，問題1で出てきた命令について文法的な説明を行います。

(1) OPEN命令

ランダムファイルをオープンする命令について，その一般形式と書き方例を次に示します。

<一般形式>

OPEN　ファイルディスクリプタ　AS　#ファイル番号

・OPEN命令は，ランダムファイルをオープンし，そのファイルに番号を付けます。

・ファイルディスクリプタで，「ドライブ番号」や「ファイル名」を指定します。

・ランダムファイルのオープンでは，シーケンシャルファイルのオープンのような「モード」の指定は行いません。

・指定したファイルが存在しない場合は，新たにファイルが作成されます。

・ファイル番号は、プログラムの中で複数のファイルを使用するときに他のファイルと区別することができるような番号を指定します。

<書き方例>

① OPEN　"2：URIAGE.MST"　AS　#1

これは，ドライブ2のフロッピーディスクに入っている「URIAGE.MST」というファイルを，ランダムファイルとしてオープンします。もし，そのフロッピーディスクに「URIAGE.MST」というファイルがない場合は，新たに作成します。

(2) FIELD#命令

ランダムファイルに格納する項目の順序や桁数を定義する命令について，その一般形式と書き方例を次に示します。

<一般形式>

FIELD#ファイル番号，桁数　AS　変数名，桁数　AS　変数名，………

・FIELD#命令は，ランダムファイルに格納するデータ(項目)の順序と桁数を定義します。

・ファイル番号は，OPEN命令で指定したファイル番号と同じ番号を指定します。

・「桁数　AS　変数名」で，一つの項目に対する定義を行います。

・桁数は，その項目に必要な桁数を指定します。

・変数名は，項目に付ける名前を指定します。変数名は，必ず「文字型」でなければなりません。ランダムファイルの項目としては，文字型しか使えません。

・1件分のデータ(レコード)として定義したすべての項目の桁数の総和が,「256桁」を超えてはいけません。なお,総和が「256桁」に満たない場合は,満たない分だけの未使用領域ができます。

<書き方例>

① FIELD#1, 1 AS FG$, 20 AS DM$

これは,ファイル番号1のファイルに格納するデータの項目の定義を次のように行っています。

FG$(1桁)	DM$(20桁)	未使用領域(235桁)

② FIELD#1, 256 AS X$

これは,誤った定義例です。

文字型変数に格納できる文字列は255文字までです。したがって,256桁の文字を一つの文字型変数に割り当てることはできません。このようなときは,次のように256桁を分割して定義します。

FIELD#1, 128 AS X1$, 128 AS X2$

(注) FIELD#命令は,定義する項目が多くなると長くなってしまい,BASICの1行の長さ(255文字)を超えることがよくあります。このようなときは,FIELD#命令を複数行に分けて定義します。(「1.3.3」参照)

(3) MKI$, MKS$, MKD$関数とCVI, CVS, CVD関数

数値データを文字型のデータに変換する関数(MKI$, MKS$, MKD$)と,文字型に変換された数値データを元に戻す関数(CVI, CVS, CVD)について説明します。

<機能>

それぞれの関数は,整数,単精度,倍精度といった数値型のデータを次のように変換します。

	変換		変換	
整数	→<MKI$>→	2桁の文字型	→<CVI>→	整数
単精度	→<MKS$>→	4桁の文字型	→<CVS>→	単精度
倍精度	→<MKD$>→	8桁の文字型	→<CVD>→	倍精度

・これらの関数は,数値型のデータをランダムファイルに登録したり,読んだりするときに使います。

（4） LSET命令とRSET命令

FIELD#命令で定義した項目へのデータセットを行う命令について，その一般形式と書き方例を次に示します。

<一般形式>

LSET　変数名$_1$＝変数名$_2$

RSET　変数名$_1$＝変数名$_2$

- LSET命令は，変数名$_1$で指定された項目に変数名$_2$のデータを左詰め（左側を揃える）でセットします。
- RSET命令は，変数名$_1$で指定された項目に変数名$_2$のデータを右詰め（右側を揃える）でセットします。
- 変数名$_1$は，FIELD#命令で定義した変数名を指定します。
- 変数名$_2$は，出力したいデータが記憶されている変数の名前を指定します。
- 変数名$_2$の代わりに文字型の定数を書くこともできます。

<書き方例>

① LSET　SFG＄＝"0"

これは，SFG＄という項目に文字定数の「0」を左詰めでセットする例です。

② RSET　NAMAE＄＝DATA＄

これは，NAMAE＄という項目にDATA＄という変数に記憶されているデータを右詰めでセットするものです。

③ LSET　ABC＄＝MKD＄(XYZ#)

これは，ABC＄という項目に，XYZ#という倍精度実数型変数を文字型に変換してから左詰めでセットするものです。

(5) PUT#命令

ランダムファイルへデータの書き込みを行う命令について，その一般形式と書き方例を次に示します。

<一般形式>

PUT#ファイル番号，レコード番号

- ファイル番号で示されるランダムファイルの中のレコード番号の位置へ,FIELD#命令で定義した各項目に入っているデータを書き込みます。
- ファイル番号は，OPEN命令で指定したファイル番号と同じ番号を指定します。
- レコード番号は，ファイル番号で指定したランダムファイルの先頭から何番目に出力するかを指定します。
- レコード番号として，「0」やフロッピーディスクに記録できる最大の件数を超える値(ミニフロッピーディスクの場合は「1265」以上，8インチ標準フロッピーディスクの場合は「3927」以上)を指定することはできません。
- レコード番号は省略することができます。この場合，直前に書き込み(または読み込み)を行ったレコード番号の次の位置へデータを書き込みます。

<書き方例>

① PUT#1，5

これは，ファイル番号1でオープンされたランダムファイルの先頭から5番目の位置に，FIELD#1で定義した各項目に入っているデータを書き込みます。

② PUT#2，I

これは，ファイル番号2でオープンされたランダムファイルの先頭から変数Iで示される位置に，FIELD#2で定義した各項目に入っているデータを書き込みます。

③ PUT#1

これは，ファイル番号1でオープンされたランダムファイルに対して，この命令以前に書き込み(または読み込み)を行ったレコード番号の次の位置にFIELD#1で定義した各項目に入っているデータを書き込みます。

1.3.3 成績データの登録方法 （ランダムファイルへのデータ登録）

前節では，成績管理ファイルの初期設定を行いましたが，まだデータを登録していないので成績管理ファイルとして使用することはできません。そこで，ここでは成績データの登録を行うプログラムを作ることにします。これをとおして，ランダムファイルへのデータの登録方法を学習して下さい。

問題 2 問題1で作成した成績管理ファイルに，次のような成績データを登録しなさい。成績データの書き込み位置は，出席番号と同じにします。

出席番号	氏　　名	国　語	数　学	理　科	社　会	英　語
1	タカハシ　タツヒコ	87	54	20	54	95
3	タナカ　ユタカ	60	80	50	50	80
5	ハラダ　イチロウ	90	89	75	54	100
7	ヤマダ　タロウ	80	91	75	88	95
9	ワタナベ　ジロウ	75	75	68	98	58
31	カワイ　ヨシエ	60	85	95	77	84
33	サトウ　ジュンコ	40	69	45	45	45
35	スズキ　ヒロコ	20	64	100	65	54
37	マツシタ　ノリコ	64	66	57	84	84
39	モリカワ　アイコ	39	45	98	100	54

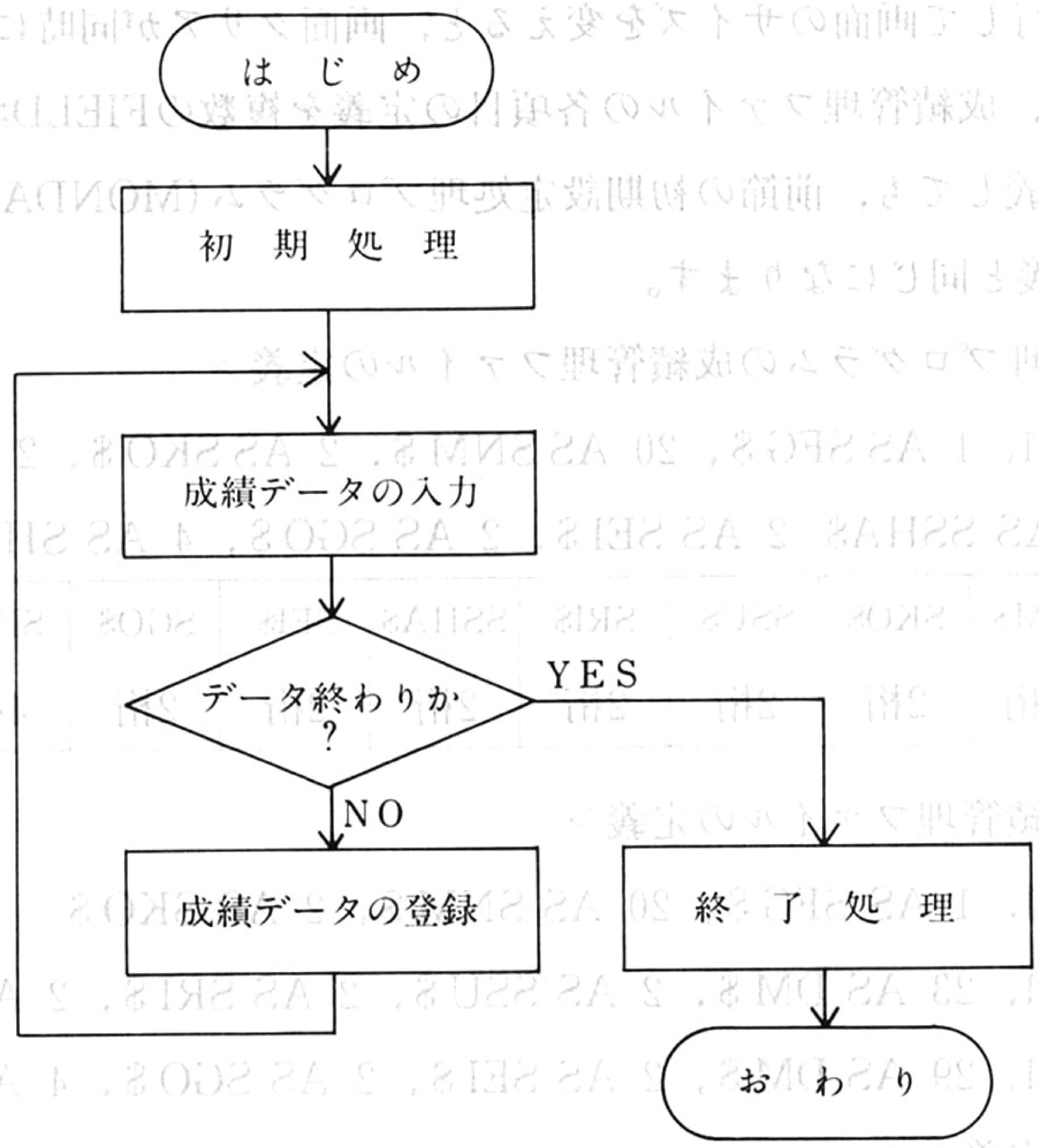
【説　明】

ランダムファイルへのデータの登録は，基本的には前節での初期設定処理と同じ要領で行うことができます。初期設定処理と違う点は，ランダムファイルへ書き込むデータが初期値ではなくキーボードやファイルなどから入力されたデータであるということです。

ここでは，問題にあるような成績データをキーボードから入力して，成績管理ファイルへ登録するプログラムを作成します。

それでは，成績管理ファイルへ成績データを登録するための処理の手順を次に示します。

＜登録処理の手順＞



【 初期処理 】

初期処理では，成績データをキーボードから入力するための準備として画面の設定とクリアを行い，成績データを登録するための準備として成績管理ファイルのオープンと項目の定義を行います。

設問 2-1 メモリーをクリアしてから，次の命令をキーインして下さい。

```
10 WIDTH 40,20:CONSOLE ,,0
20 OPEN "2:SEISEK.MST" AS #1
30 FIELD#1,1 AS SFG$,20 AS SNM$,2 AS SKO$
40 FIELD#1,23 AS DM$,2 AS SSU$,2 AS SRI$,2 AS SSHA$
50 FIELD#1,29 AS DM$,2 AS SEI$,2 AS SGO$,4 AS SHE$
```

〔キーイン後の説明〕

① 行番号10は，画面を40×20のサイズにした後，ファンクションキーラベルの表示を行わないようにしています。

ここで入力するデータの項目は，40×20の画面サイズで十分間に合います。そこで，画面の文字が大きくて見やすいということも考慮して，画面のサイズを40×20に設定しています。

画面処理を行うプログラムでは，通常初期処理で画面のクリアを行います。なお，WIDTH命令を実行して画面のサイズを変えると，画面クリアが同時に行なわれます。

② 行番号30～50は，成績管理ファイルの各項目の定義を複数のFIELD#命令で行っています。このように定義しても，前節の初期設定処理プログラム(MONDAI．1)の行番号20のFIELD#命令の定義と同じになります。

<初期設定処理プログラムの成績管理ファイルの定義>

20 FIELD#1, 1 AS SFG＄, 20 AS SNM＄, 2 AS SKO＄, 2 AS SSU＄, 2 AS SRI＄, 2 AS SSHA$ 2 AS SEI＄, 2 AS SGO＄, 4 AS SHE＄

SFG$	SNM$	SKO$	SSU$	SRI$	SSHA$	SEI$	SGO$	SHE$	未定義
1桁	20桁	2桁	2桁	2桁	2桁	2桁	2桁	4桁	219桁

<ここでの成績管理ファイルの定義>

30 FIELD#1, 1 AS SFG＄, 20 AS SNM＄, 2 AS SKO＄

40 FILED#1, 23 AS DM＄, 2 AS SSU＄, 2 AS SRI＄, 2 AS SSHA＄

50 FIELD#1, 29 AS DM＄, 2 AS SEI＄, 2 AS SGO＄, 4 AS SHE＄

・行番号30の定義

SFG$	SNM$	SKO$	未　定　義
1桁	20桁	2桁	233桁

・行番号40の定義

DM$	SSU$	SRI$	SSHA$	未　定　義
23桁	2桁	2桁	2桁	227桁

・行番号50の定義

DM$	SEI$	SGO$	SHE$	未定義
29桁	2桁	2桁	4桁	219桁

※ このように定義することによって，次のように(初期設定処理プログラムと同じ)定義をしたことになります。

SFG$	SNM$	SKO$	SSU$	SRI$	SSHA$	SEI$	SGO$	SHE$	未定義
1桁	20桁	2桁	2桁	2桁	2桁	2桁	2桁	4桁	219桁

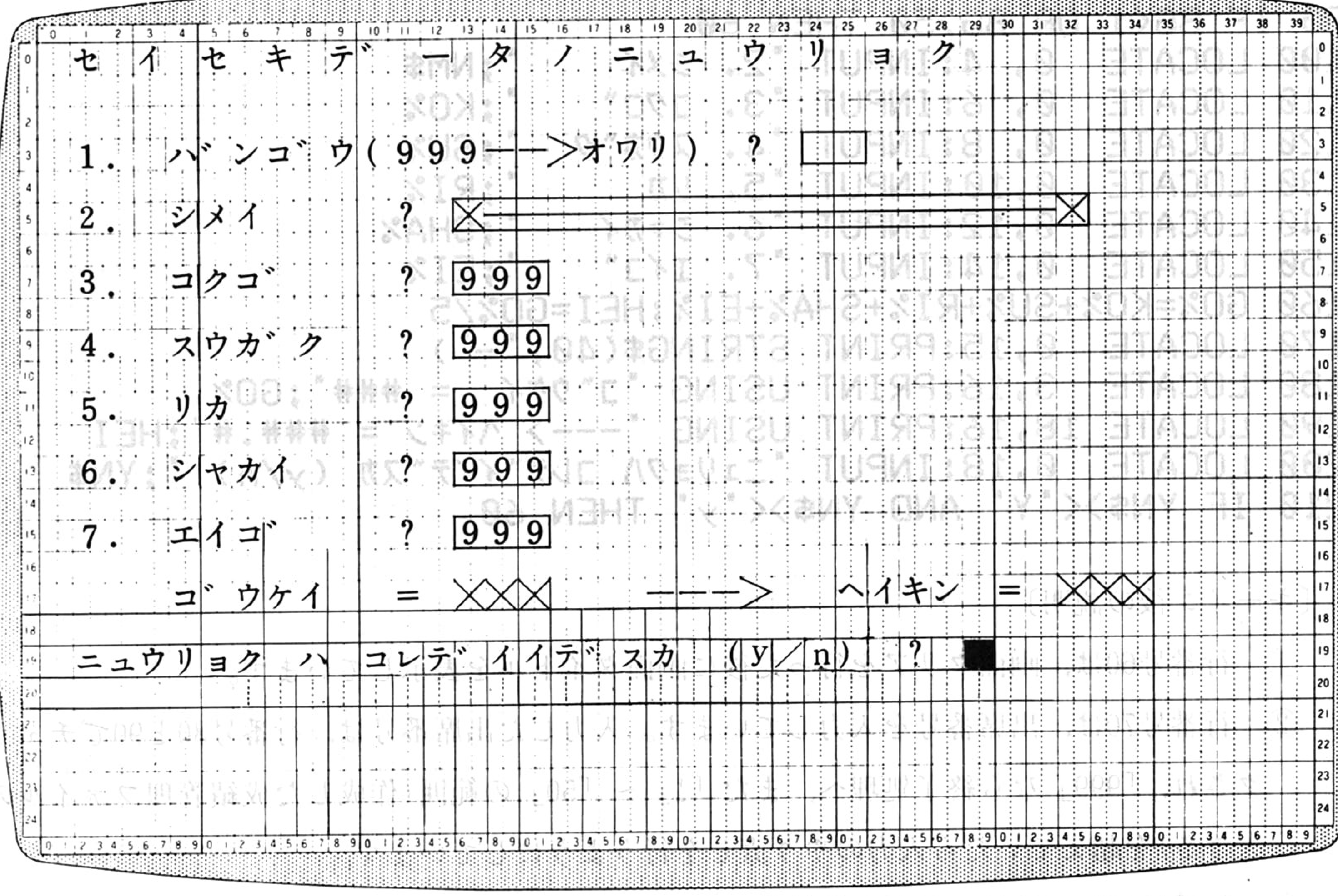
このように，複数のFIELD#命令で項目の定義をするやり方は，項目の数が多くて一行(255字)で収まらない場合や，画面の一行(80字)に収まらずに見づらい場合に便利です。

なお，ここで桁合わせのために使っているDM＄という項目はダミーの指定ですが，後から定義した行番号50のDM＄(29桁)は有効です。したがって，データの先頭から29桁分(SFG＄，SNM＄，SKO＄，SSU＄，SRI＄，SSHA＄)は，DM＄という項目にまとめて入っていることになります。

【成績データの入力】

問題にあるような成績データを入力するときの画面を次のように想定(「画面設計」といいます)します。

＜入力画面＞



上の画面を使って成績データをキーボードから入力するプログラムの処理概要について次に示します。

＜処理概要＞

- 出席番号として「999」が入力されたときは，処理を終了します。
- 各教科の得点を入力したら，合計と平均を求めて表示します。
- 入力に間違いがあったら，入力をやり直します。

設 問 2-2 成績データの入力を行うために，次の命令をキーインして下さい。

```
60 CLS:LOCATE  0, 0:PRINT "ｾ ｲ ｾ ｷ ﾃﾞ ｰ ﾀ ﾉ ﾆ ｭ ｳ ﾘ ｮ ｸ"
70 LOCATE  0, 2:INPUT "1. ﾊﾞﾝｺﾞｳ(999-->ｵﾜﾘ) ";BG
80 IF BG=999 THEN 330
90 IF BG<1 OR BG>50 THEN 60
100 LOCATE  0, 4:INPUT "2. ｼﾒｲ      ";NM$
110 LOCATE  0, 6:INPUT "3. ｺｸｺﾞ     ";KO%
120 LOCATE  0, 8:INPUT "4. ｽｳｶﾞｸ    ";SU%
130 LOCATE  0,10:INPUT "5. ﾘｶ       ";RI%
140 LOCATE  0,12:INPUT "6. ｼｬｶｲ     ";SHA%
150 LOCATE  0,14:INPUT "7. ｴｲｺﾞ     ";EI%
160 GO%=KO%+SU%+RI%+SHA%+EI%:HEI=GO%/5
170 LOCATE  0,15:PRINT STRING$(40,"-")
180 LOCATE  3,16:PRINT USING "ｺﾞｳｹｲ  = ###";GO%
190 LOCATE 18,16:PRINT USING "---> ﾍｲｷﾝ = ###.#";HEI
200 LOCATE  0,18:INPUT "ﾆｭﾘｮｸﾊ ｺﾚﾃﾞｲｲﾃﾞｽｶ (y/n) ";YN$
210 IF YN$><"Y" AND YN$><"y" THEN 60
```

〔キーイン後の説明〕

① 行番号60は，画面クリアを行った後に画面タイトルを表示しています。

② 行番号70は，出席番号を入力しています。入力した出席番号は，行番号80と90でチェックされ，「999」なら終了処理へ，また「1」～「50」の範囲(作成した成績管理ファイルの大きさ)を超えていたら入力のやり直しを行うようにしています。

③ 行番号100～150は，氏名と各教科の得点の入力を行っています。

④ 行番号160～190は，合計と平均を求めてその表示を行っています。

⑤ 行番号200～210は，入力項目に間違いがないかどうかを確認し，間違いがあった場合は入力をやり直すことができるようにしています。

【 成績データの登録 】

キーボードから入力したデータを，FIELD#命令で定義した各項目へセットして成績管理ファイルへ登録します。

設問 2-3 成績データのセットと登録を行う次の命令をキーインして下さい。

```
220 LSET SFG$="1"
230 LSET SNM$=NM$
240 LSET SKO$=MKI$(KO%)
250 LSET SSU$=MKI$(SU%)
260 LSET SRI$=MKI$(RI%)
270 LSET SSHA$=MKI$(SHA%)
280 LSET SEI$=MKI$(EI%)
290 LSET SGO$=MKI$(GO%)
300 LSET SHE$=MKS$(HEI)
310 PUT#1,BG
320 GOTO 60
```

〔キーイン後の説明〕

① 行番号220で，識別コードに文字定数の「1」をセットしています。これによって，データが入っているレコードの識別コードは「1」，データが入っていないレコードの識別コードは「0」になっていることになります。

② 行番号230～300は，入力した成績データと計算した合計と平均とをFIELD#命令で定義した各項目へセットしています。

③ 行番号310は，入力した出席番号(変数BG)の位置にデータを登録します。

④ 行番号320は，次のデータを入力するために行番号60へ戻ります。

設問 2-4 終了処理を行う次の命令をキーインして下さい。

```
330 CLOSE
340 WIDTH 80,25
350 END
```

〔キーイン後の説明〕

① 行番号330は，成績管理ファイルのクローズを行います。

② 行番号340は，画面を40×20のサイズから80×25のサイズにしています。

【 プログラムの確認 】

設問 2-5 正しくキーインできているかどうかを確認するために，キーインしたプログラムのリストをプリンタに印字して下さい。

＜確認用リスト＞

```
10 WIDTH 40,20:CONSOLE ,,0
20 OPEN "2:SEISEK.MST" AS #1
30 FIELD#1,1 AS SFG$,20 AS SNM$,2 AS SKO$
40 FIELD#1,23 AS DM$,2 AS SSU$,2 AS SRI$,2 AS SSHA$
50 FIELD#1,29 AS DM$,2 AS SEI$,2 AS SGO$,4 AS SHE$
60 CLS:LOCATE  0, 0:PRINT "セ イ セ キ テ゛ ー タ ノ ニ ュ ウ リ ョ ク"
70 LOCATE  0, 2:INPUT "1. ハ゛ンコ゛ウ(999-->オワリ) ";BG
80 IF BG=999 THEN 330
90 IF BG<1 OR BG>50 THEN 60
100 LOCATE  0, 4:INPUT "2. シメイ      ";NM$
110 LOCATE  0, 6:INPUT "3. コクコ゛    ";KO%
120 LOCATE  0, 8:INPUT "4. スウカ゛ク   ";SU%
130 LOCATE  0,10:INPUT "5. リカ       ";RI%
140 LOCATE  0,12:INPUT "6. シャカイ    ";SHA%
150 LOCATE  0,14:INPUT "7. エイコ゛    ";EI%
160 GO%=KO%+SU%+RI%+SHA%+EI%:HEI=GO%/5
170 LOCATE  0,15:PRINT STRING$(40,"-")
180 LOCATE  3,16:PRINT USING "コ゛ウケイ  = ###";GO%
190 LOCATE 18,16:PRINT USING "---> ヘイキン = ###.#";HEI
200 LOCATE  0,18:INPUT "ニュリョクハ コレテ゛イイテ゛スカ (y/n) ";YN$
210 IF YN$><"Y" AND YN$><"y" THEN 60
220 LSET SFG$="1"
230 LSET SNM$=NM$
240 LSET SKO$=MKI$(KO%)
250 LSET SSU$=MKI$(SU%)
260 LSET SRI$=MKI$(RI%)
270 LSET SSHA$=MKI$(SHA%)
280 LSET SEI$=MKI$(EI%)
290 LSET SGO$=MKI$(GO%)
300 LSET SHE$=MKS$(HEI)
310 PUT#1,BG
320 GOTO 60
330 CLOSE
340 WIDTH 80,25
350 END
```

【 プログラムの実行 】

設問 2-6 正しくキーインできていたら，プログラムを実行して問題 2 に示してあった次のような成績データを登録して下さい。

＜登録する成績データ＞

出席番号	氏　　名	国　語	数　学	理　科	社　会	英　語
1	タカハシ　タツヒコ	87	54	20	54	95
3	タナカ　ユタカ	60	80	50	50	80
5	ハラダ　イチロウ	90	89	75	54	100
7	ヤマダ　タロウ	80	91	75	88	95
9	ワタナベ　ジロウ	75	75	68	98	58
31	カワイ　ヨシエ	60	85	95	77	84
33	サトウ　ジュンコ	40	69	45	45	45
35	スズキ　ヒロコ	20	64	100	65	54
37	マツシタ　ノリコ	64	66	57	84	84
39	モリカワ　アイコ	39	45	98	100	54

〔実行時の注意〕

① 「1. バンゴウ」に対しては，出席番号を入力します。出席番号として「999」を入力すると処理は終了します。

② 2件目以降のデータの入力は，1件目のデータの入力と同じような方法で順次入力します。

＜実行例＞

```
セ イ セ キ デ゛ー タ ノ ニ ュ ウ リ ョ ク

1. ハ゛ンコ゛ウ(999-->オワリ) ? 1

2. シメイ      ? タカハシ タツヒコ

3. コクコ゛    ? 87

4. スウカ゛ク  ? 54

5. リカ        ? 20

6. シャカイ    ? 54

7. エイコ゛    ? 95
-----------------------------------------------
   コ゛ウケイ  = 310    ---> ヘイキン =   62.0
```

(つづく)

```
ﾆｭｳﾘｮｸﾊ ｺﾚﾃﾞｲｲﾃﾞｽｶ (y/n) ? y


6. ｼｬｶｲ   ? 100

7. ｴｲｺﾞ   ? 54
--------------------------------------------------
   ｺﾞｳｹｲ  = 336    ---> ﾍｲｷﾝ =  67.2

ﾆｭｳﾘｮｸﾊ ｺﾚﾃﾞｲｲﾃﾞｽｶ (y/n) ? y
```

〔実行後の説明〕

① 入力したデータは，フロッピーディスクの「SEISEK.MST」というランダムファイルに次のように記録されているはずです。

レコード番号	20桁 (氏名)	2桁 (国語)	2桁 (数学)	2桁 (理科)	2桁 (社会)	2桁 (英語)	2桁 (合計)	4桁 (平均)
1	タカハシ　タツヒコ	87	54	20	54	95	310	62.0
2	空　白	0	0	0	0	0	0	0.0
3	タナカ　ユタカ	60	80	50	50	80	320	64.0
4	空　白	0	0	0	0	0	0	0.0
5	ハラダ　イチロウ	90	89	75	54	100	408	81.6
6	空　白	0	0	0	0	0	0	0.0
7	ヤマダ　タロウ	80	91	75	88	95	429	85.8
8	空　白	0	0	0	0	0	0	0.0
9	ワタナベ　ジロウ	75	75	68	98	58	374	74.8
10 ≀ 30	空　白	0	0	0	0	0	0	0.0
31	カワイ　ヨシエ	60	85	95	77	84	401	80.2
32	空　白	0	0	0	0	0	0	0.0
33	サトウ　ジュンコ	40	69	45	45	45	244	48.8
34	空　白	0	0	0	0	0	0	0.0
35	スズキ　ヒロコ	20	64	100	65	54	303	60.6
36	空　白	0	0	0	0	0	0	0.0
37	マツシタ　ノリコ	64	66	57	84	84	355	71.0
38	空　白	0	0	0	0	0	0	0.0

レコード番号	20桁（氏名）	2桁（国語）	2桁（数学）	2桁（理科）	2桁（社会）	2桁（英語）	2桁（合計）	4桁（平均）
39	モリカワ　アイコ	39	45	98	100	54	336	67.2
40〜50	空　白	0	0	0	0	0	0	0.0

② 正しく記録できているかどうかについては，次節の成績一覧表で確認します。

③ このプログラムは，出席番号をレコード番号として利用しました。ところが，出席番号がこのようにうまくレコード番号として利用できない場合があります。この場合は，出席番号もデータ項目としてファイルへ登録します。そして，出席番号とは別にレコード番号を設定して，ランダムファイルへデータを登録するようにします。

設問 2-7 プログラムをドライブ 2 のフロッピーディスクへ「MONDAI.2」という名前で保存して下さい。

<実行結果例>

```
save "2:MONDAI.2"
Ok
```

1.3.4　わかりやすいプログラムの作成

プログラムは，後で機能を修正したり拡張したりすることがよくあります。そこで，後でプログラムを見直したときに(修正や拡張のために)，内容がわかりやすく修正もしやすいようにプログラムを作ります。

これは，一般にBASICのプログラムでは，作業の手順に沿ってその処理命令を並べるだけのものが多く，後から見直したときにわかりにくいということがよくあるからです。

「1.3.3」で作成したプログラムも，「成績データの登録」という作業の手順に沿って順番に処理命令を並べて組まれています。このようなプログラムは，作成するときは容易なのですが，しばらくしてから見直すと，どういう手順で何をやっているかがわかりにくくなります。

これは，プログラムの流れを制御する命令と処理を行う命令とが混在しているために，プログラムの流れを理解するのに最初から最後までプログラムを見直さなければならないからです。

そこで，プログラムの流れをわかりやすくするために，流れを制御する命令と処理を行う命令とがはっきりと区別できるようなプログラムを作成することにします。

次のプログラムを見て下さい。

```
100 ' ｾｲｾｷﾃﾞｰﾀ ﾆｭｳﾘｮｸ        (MONDAI.2-X)
110 *HAJIME:' <<< ｼｮｷ ｼｮﾘ >>>
120  WIDTH 40,20:CONSOLE ,,0
130  OPEN "2:SEISEK.MST" AS #1
140  FIELD#1,1 AS SFG$,20 AS SNM$,2 AS SKO$
150  FIELD#1,23 AS DM$,2 AS SSU$,2 AS SRI$,2 AS SSHA$
160  FIELD#1,29 AS DM$,2 AS SEI$,2 AS SGO$,4 AS SHE$
170 *MAIN:' <<< ﾒｲﾝ ﾙｰﾁﾝ >>>
180  GOSUB *NYURYOKU
190  IF BG=999 THEN *OWARI
200  GOSUB *DATA.PUT
210  GOTO *MAIN
220 *OWARI:' <<< ｼｭｳﾘｮｳ ｼｮﾘ >>>
230  CLOSE
240  WIDTH 80,25
250  END
1000 '
1010 *NYURYOKU:' <<< ﾃﾞｰﾀﾆｭｳﾘｮｸ ｼｮﾘ >>>
1020  CLS:LOCATE  0, 0:PRINT "ｾ ｲ ｾ ｷ ﾃﾞ ｰ ﾀ ﾉ ﾆ ｭ ｳ ﾘ ｮ ｸ"
1030  LOCATE  0, 2:INPUT "1. ﾊﾞﾝｺﾞｳ(999-->ｵﾜﾘ) ";BG
1040  IF BG=999 THEN 1180
1050  IF BG<1 OR BG>50 THEN *NYURYOKU
1060  LOCATE  0, 4:INPUT "2. ｼﾒｲ     ";NM$
1070  LOCATE  0, 6:INPUT "3. ｺｸｺﾞ    ";KO%
1080  LOCATE  0, 8:INPUT "4. ｽｳｶﾞｸ   ";SU%
1090  LOCATE  0,10:INPUT "5. ﾘｶ     ";RI%
1100  LOCATE  0,12:INPUT "6. ｼｬｶｲ    ";SHA%
1110  LOCATE  0,14:INPUT "7. ｴｲｺﾞ    ";EI%
1120  GO%=KO%+SU%+RI%+SHA%+EI%:HEI=GO%/5
1130  LOCATE  0,15:PRINT STRING$(40,"-")
1140  LOCATE  3,16:PRINT USING "ｺﾞｳｹｲ  = ###";GO%
```

（つづく）

```
1150  LOCATE 18,16:PRINT USING "---> ﾍｲｷﾝ = ###.#";HEI
1160  LOCATE  0,18:INPUT "ﾆｭﾘｮｸﾊ ｺﾚﾃﾞｲｲﾃﾞｽｶ (y/n) ";YN$
1170  IF YN$><"Y" AND YN$><"y" THEN *NYURYOKU
1180  RETURN
2000 '
2010 *DATA.PUT:' <<< ﾃﾞｰﾀﾉｾｯﾄ ﾄ ｶｷｺﾐ >>>
2020  LSET SFG$="1"
2030  LSET SNM$=NM$
2040  LSET SKO$=MKI$(KO%)
2050  LSET SSU$=MKI$(SU%)
2060  LSET SRI$=MKI$(RI%)
2070  LSET SSHA$=MKI$(SHA%)
2080  LSET SEI$=MKI$(EI%)
2090  LSET SGO$=MKI$(GO%)
2100  LSET SHE$=MKS$(HEI)
2110  PUT#1,BG
2120  RETURN
```

このプログラムは，成績データの入力処理と登録処理をサブルーチン化してその実行をメインルーチン(＊MAIN)で制御するようにしています。

前のプログラム(MONDAI.2)と比べてみて下さい。処理の流れが行番号170～210のメインルーチンを見るだけでわかります。また，各処理の機能ごとにサブルーチン化されているために，ここからここまでで何をやっているかといったことがはっきりしています。

このように，サブルーチン化を行うことによって，プログラムはかなりわかりやすいものになります。

※　このプログラムでは，サブルーチンの区切りをはっきりさせるために，ラベル名以外の行の先頭に一文字の空白をいれて「＊」を目立たせるようにしています。

設問 2-8　いまメモリーに入っているプログラムを，次の手順に従ってサブルーチン化します。

〔手順〕

① 次の命令をキーインして下さい。

```
2 ' ｾｲｾｷﾃﾞｰﾀ ﾆｭｳﾘｮｸ        (MONDAI.2-X)
4 *HAJIME:' <<< ｼｮｷ ｼｮﾘ >>>
51 *MAIN:' <<< ﾒｲﾝ ﾙｰﾁﾝ >>>
52 GOSUB *NYURYOKU
53 IF BG=999 THEN *OWARI
54 GOSUB *DATA.PUT
55 GOTO *MAIN
56 '
```
(つづく)

```
57 *NYURYOKU:' <<< デ゛ータニュウリョク ショリ >>>
80 IF BG=999 THEN 212
90 IF BG<1 OR BG>50 THEN *NYURYOKU
210 IF YN$><"Y" AND YN$><"y" THEN *NYURYOKU
212 RETURN
214 '
216 *DATA.PUT:' <<< デ゛ータノセット ト カキコミ >>>
320 RETURN
322 *OWARI:' <<< シュウリョウ ショリ >>>
```

② 行番号を次のように付け直して下さい。

```
RENUM 100
```

③ 行番号540以降を表示して，その行番号を次のように変更した後，行番号540以降を削除して下さい。

〔修正要領説明〕

既にメモリーに入っているプログラムの命令と同じものをキーインする場合，行番号だけを変更して ⏎ キーを押すことによりコピーすることができます。

ここでは，行番号540～570までの命令を新しく行番号212～218へコピーします。修正の手順としては，まず行番号540～570までの命令を画面に表示した後カーソルを行番号の位置へ移動して新しくコピーする行番号に変更して ⏎ キーを押します。

なお，ここでは行番号540～570は不要になるので削除します。

```
540 → 212
550 → 214
560 → 216
570 → 218
DELETE 540-570
```

④ 再度，行番号を次のように付け直して下さい。

```
RENUM 100
RENUM 1000,260
RENUM 2000,1190
```

⑤ プログラムを見やすくするためラベル名以外の各行の先頭に1文字の空白を入れて下さい。

設問 2-9 プログラムのリストをプリンタに印字し，修正結果を確認して下さい。

＜確認用リスト＞

```
100 ' ｾｲｾｷﾃﾞｰﾀ ﾆｭｳﾘｮｸ        (MONDAI.2-X)
110 *HAJIME:' <<< ｼｮｷ ｼｮﾘ >>>
120  WIDTH 40,20:CONSOLE ,,0
130  OPEN "2:SEISEK.MST" AS #1
140  FIELD#1,1 AS SFG$,20 AS SNM$,2 AS SKO$
150  FIELD#1,23 AS DM$,2 AS SSU$,2 AS SRI$,2 AS SSHA$
160  FIELD#1,29 AS DM$,2 AS SEI$,2 AS SGO$,4 AS SHE$
170 *MAIN:' <<< ﾒｲﾝ ﾙｰﾁﾝ >>>
180  GOSUB *NYURYOKU
190  IF BG=999 THEN *OWARI
200  GOSUB *DATA.PUT
210  GOTO *MAIN
220 *OWARI:' <<< ｼｭｳﾘｮｳ ｼｮﾘ >>>
230  CLOSE
240  WIDTH 80,25
250  END
1000 '
1010 *NYURYOKU:' <<< ﾃﾞｰﾀﾆｭｳﾘｮｸ ｼｮﾘ >>>
1020  CLS:LOCATE  0, 0:PRINT "ｾ ｲ ｾ ｷ ﾃﾞ ｰ ﾀ ﾉ ﾆ ｭ ｳ ﾘ ｮ ｸ"
1030  LOCATE  0, 2:INPUT "1. ﾊﾞﾝｺﾞｳ(999-->ｵﾜﾘ) ";BG
1040  IF BG=999 THEN 1180
1050  IF BG<1 OR BG>50 THEN *NYURYOKU
1060  LOCATE  0, 4:INPUT "2. ｼﾒｲ     ";NM$
1070  LOCATE  0, 6:INPUT "3. ｺｸｺﾞ    ";KO%
1080  LOCATE  0, 8:INPUT "4. ｽｳｶﾞｸ   ";SU%
1090  LOCATE  0,10:INPUT "5. ﾘｶ      ";RI%
1100  LOCATE  0,12:INPUT "6. ｼｬｶｲ    ";SHA%
1110  LOCATE  0,14:INPUT "7. ｴｲｺﾞ    ";EI%
1120  GO%=KO%+SU%+RI%+SHA%+EI%:HEI=GO%/5
1130  LOCATE  0,15:PRINT STRING$(40,"-")
1140  LOCATE  3,16:PRINT USING "ｺﾞｳｹｲ  = ###";GO%
1150  LOCATE 18,16:PRINT USING "---> ﾍｲｷﾝ = ###.#";HEI
1160  LOCATE  0,18:INPUT "ﾆｭﾘｮｸﾊ ｺﾚﾃﾞｲｲﾃﾞｽｶ (y/n) ";YN$
```

(つづく)

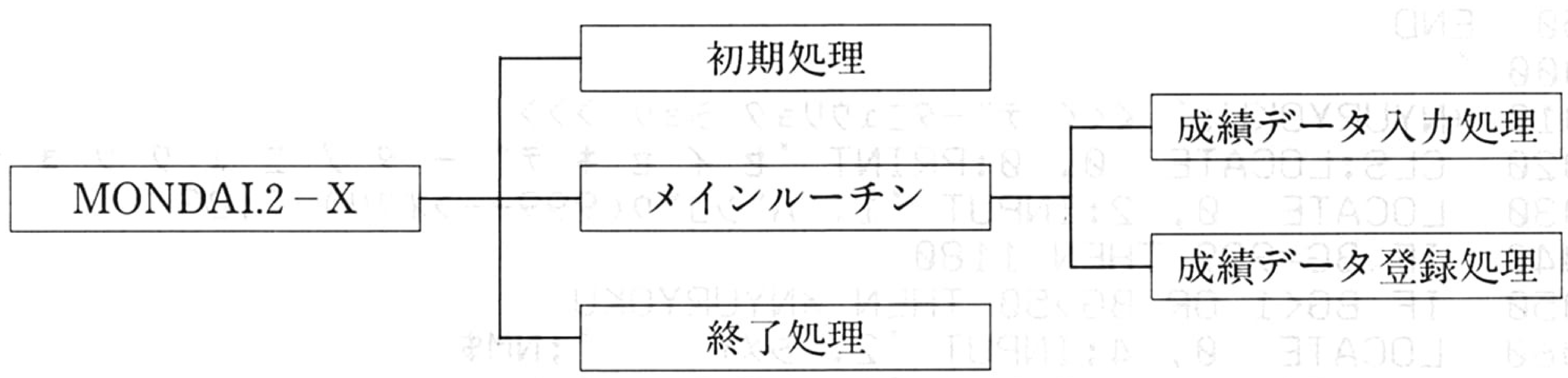
```
1170  IF YN$><"Y" AND YN$><"y" THEN *NYURYOKU
1180  RETURN
2000 '
2010 *DATA.PUT:' <<< ﾃﾞｰﾀﾉｾｯﾄ ﾄ ｶｷｺﾐ >>>
2020  LSET SFG$="1"
2030  LSET SNM$=NM$
2040  LSET SKO$=MKI$(KO%)
2050  LSET SSU$=MKI$(SU%)
2060  LSET SRI$=MKI$(RI%)
2070  LSET SSHA$=MKI$(SHA%)
2080  LSET SEI$=MKI$(EI%)
2090  LSET SGO$=MKI$(GO%)
2100  LSET SHE$=MKS$(HEI)
2110  PUT#1,BG
2120  RETURN
```

設問 2-10 正しくキーインできていたら，プログラムをドライブ 2 のフロッピーディスクへ「MONDAI.2－X」という名前で保存して下さい。

＜実行結果例＞

```
save "2:MONDAI.2-X"
Ok
```

※　ここで作成したプログラムの構造を次に示します。



1.3.5 成績一覧表を印字する（ランダムファイルの順読み込み）

「1.3.3 成績データの登録方法」で成績管理ファイルに登録したデータを順番に読み込んで，成績一覧表を作成します。

問題 3 問題2で登録した成績データを順に読み込んで，次のような成績一覧表を作りなさい。

中 間 考 査 成 績 一 覧 表

番号	氏　　名	国語	数学	理科	社会	英語	合計	平 均
1	ﾀｶﾊｼ ﾀﾂﾋｺ	87	54	20	54	95	310	62.0
3	ﾀﾅｶ ﾕﾀｶ	60	80	50	50	80	320	64.0
5	ﾊﾗﾀﾞ ｲﾁﾛｳ	90	89	75	54	100	408	81.6
7	ﾔﾏﾀﾞ ﾀﾛｳ	80	91	75	88	95	429	85.8
9	ﾜﾀﾅﾍﾞ ｼﾞﾛｳ	75	75	68	98	58	374	74.8
31	ｶﾜｲ ﾖｼｴ	60	85	95	77	84	401	80.2
33	ｻﾄｳ ｼﾞｭﾝｺ	40	69	45	45	45	244	48.8
35	ｽｽﾞｷ ﾋﾛｺ	20	64	100	65	54	303	60.6
37	ﾏﾂｼﾀ ﾉﾘｺ	64	66	57	84	84	355	71.0
39	ﾓﾘｶﾜ ｱｲｺ	39	45	98	100	54	336	67.2
10人	ク ラ ス 平 均	61.5	71.8	68.3	71.5	74.9	348.0	69.6

【説　明】

ランダムファイルからのデータ入力は，レコード番号を指定することによってどの位置のデータも自由に読み込めるのが特長ですが，ここではシーケンシャルファイルのときと同じようにファイルの先頭から順番にデータを読み込んで処理を行います。

また，帳表を出力するプログラムでは，作成前に必ず帳表のイメージを設計しておく必要があります。そこで，初めに見出しの印字位置や明細の印字位置の設計(「帳表フォーマットの設計」といいます)を行います。

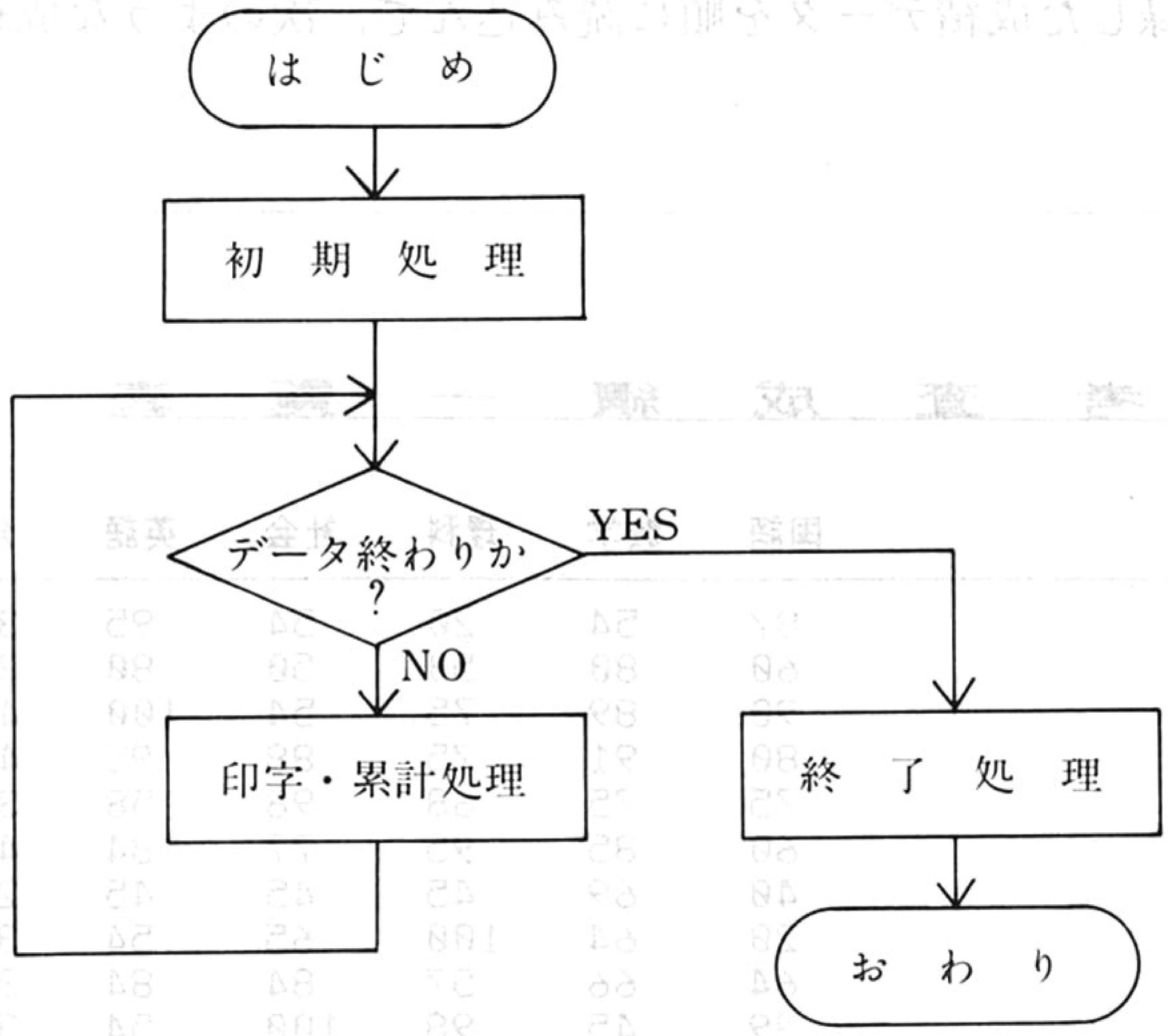
それでは，成績一覧表を出力するための処理概要とフローチャートおよび帳表フォーマットを次に示します。

＜処理概要＞

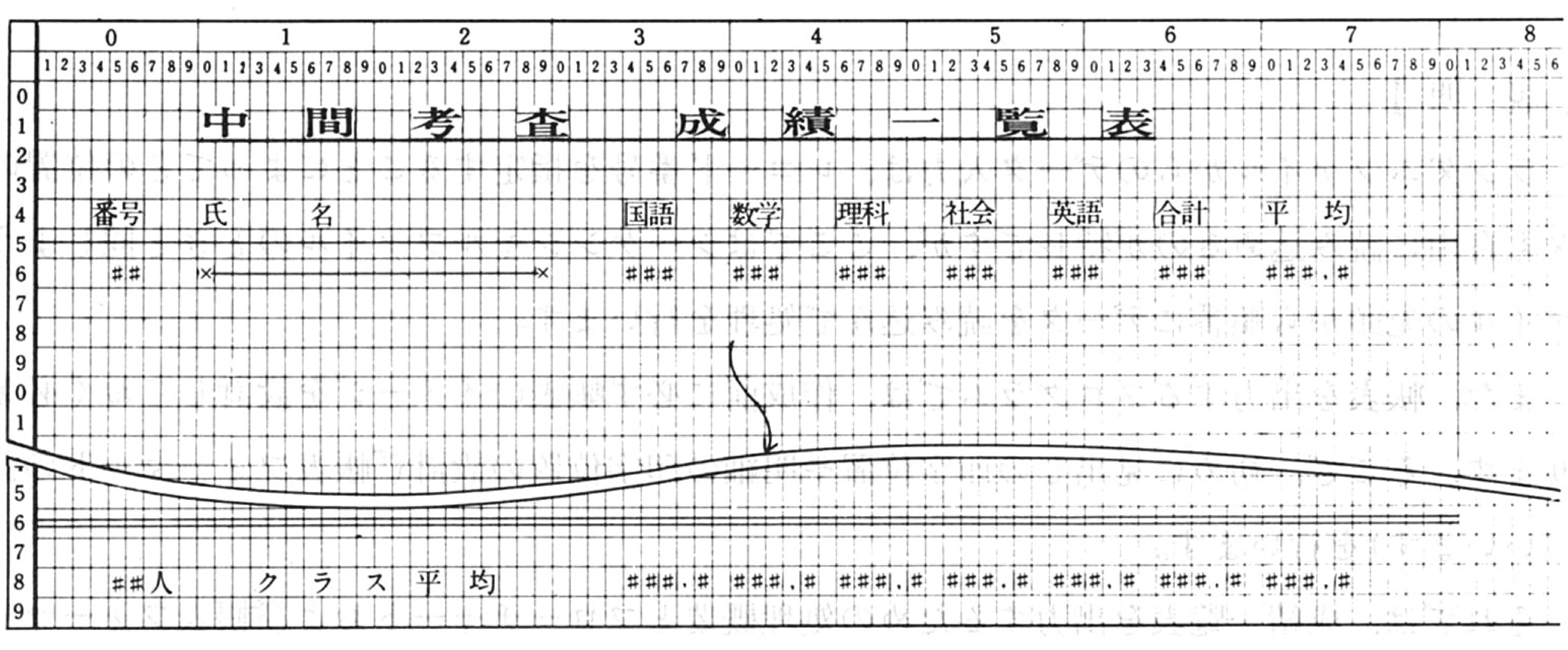
- 成績管理ファイルから成績データを1件ずつ読み込み，識別コードを判定します。
- 識別コードが「1」のデータは，その明細を印字し累計をとるようにします。それ以外のデータは読み飛ばします。

・成績管理ファイルに入っているデータを全部読み終わったら，累計から平均を求めて印字し処理を終了します。

＜成績一覧表を出力するプログラムのフローチャート＞



＜成績一覧表の帳表フォーマット＞



※　帳表フォーマットを設計する用紙は，プリントレイアウト用紙(またはスペーシングチャート)と呼ばれるものです。この用紙の1目盛りごとの枠は，プリンタで印字する文字の1文字分と対応します。

なお，漢字は普通の文字の1.5倍の大きさで印字されることに注意して下さい。

【 プログラムの作成 】

初期処理として，成績管理ファイルのオープン，フィールドの定義，見出しの印字などを行います。

プログラムのキーインを行う前に，メモリーに残っている成績データの登録プログラムを見て下さい。このうち，初期処理の部分と終了処理の部分は同じように使えるので利用することにします。

設問 3-1 いまメモリーに入っているプログラムを利用するために,不要な行を削除してリストを表示して下さい。

<削除する行>

```
DELETE 180-210
DELETE 1000-2120
```

<削除後のプログラムリスト>

```
100 ' セイセキイチランヒョウ インジ"      (MONDAI.3)
110 *HAJIME:' <<< ショキ ショリ >>>
120  WIDTH 40,20:CONSOLE ,,0
130  OPEN "2:SEISEK.MST" AS #1
140  FIELD#1,1 AS SFG$,20 AS SNM$,2 AS SKO$
150  FIELD#1,23 AS DM$,2 AS SSU$,2 AS SRI$,2 AS SSHA$
160  FIELD#1,29 AS DM$,2 AS SEI$,2 AS SGO$,4 AS SHE$
170 *MAIN:' <<< メイン ルーチン >>>
220 *OWARI:' <<< シュウリョウ ショリ >>>
230  CLOSE
240  WIDTH 80,25
250  END
```

〔削除後の説明〕

① ここでの処理は，ファイルに登録した成績データを読み込んで一覧表を印字することです。したがって，成績データをキーボードから入力する処理(＊NYURYOKU)と成績管理ファイルへ登録する処理(＊DATA.PUT)が不要になるので削除しました。

設問 3-2 次の命令を追加して下さい。

```
100 ' セイセキイチランヒョウ インジ"       (MONDAI.3)
165  GOSUB *MIDASHI
180  FOR BG=1 TO LOF(1)
190    GOSUB *MEISAI
200  NEXT BG
225  GOSUB *CLASS.HEIKIN
```

〔追加後の説明〕

① 見出しを印字する処理ルーチン(＊MIDASHI)を初期処理の中に入れます。

② 明細印字を行う処理を，成績管理ファイルに入っているデータの件数分だけ行います。

③ 変数BGは，出席番号(＝レコード番号)の指定に使います。

④ LOF(1)は，ファイル番号1のファイル(成績管理ファイル)に入っているデータの中で一番大きなレコード番号を求める関数です。(【 命令のまとめ 】 参照)

⑤ クラス平均を求めて印字する処理ルーチン(＊CLASS.HEIKIN)を終了処理の中に入れます。

設問 3-3 見出しを印字する処理ルーチン(＊MIDASHI)を追加して下さい。

＜追加する処理ルーチン＞

```
1000 '
1010 *MIDASHI:' <<< ミダ"シ インジ" セイキ"ョ ルーチン >>>
1020  RESTORE 10010
1030  LPRINT CHR$(&HC);:LPRINT                  : ' &hC (カイヘ°ージ")
1040  SP=9:KA=1:UL=1:KJ=34:GOSUB *MID.INJI
1050  LPRINT:LPRINT CHR$(&H1F);CHR$(&H10+2);
1060  SP=3 :KJ=4:GOSUB *MID.INJI              : ' SP (スヘ°ース ノ カズ")
1070  SP=3 :KJ=12:GOSUB *MID.INJI              : ' KJ (カンジ"コード"ノカズ")
1080  SP=15:KJ=4:GOSUB *MID.INJI
1090  SP=3 :KJ=4:GOSUB *MID.INJI
1100  SP=3 :KJ=4:GOSUB *MID.INJI
1110  SP=3 :KJ=4:GOSUB *MID.INJI
1120  SP=3 :KJ=4:GOSUB *MID.INJI
1130  SP=3 :KJ=4:GOSUB *MID.INJI
1140  SP=3 :KJ=6:GOSUB *MID.INJI
1150  LPRINT:LPRINT STRING$(79,"ー")
1160  RETURN
2000 '
2010 *MID.INJI:' <<< ミダ"シ インジ" ルーチン >>>
2020  IF SP><0 THEN LPRINT SPC(SP);
2030  LPRINT CHR$(&H1B);"K";                    : ' ESC-K (カンジ"モード")
2040  IF KA=1 THEN LPRINT CHR$(&HE);           : ' &HE (カクダ"イ)
2050  IF UL=1 THEN LPRINT CHR$(&H1B);"X";: ' ESC-X (アンダ"ーライン)
2060  FOR J=1 TO KJ
```

(つづく)

```
2070     READ X$                                  :' カンジ"コード" ヨミコミ
2080     LPRINT CHR$(VAL("&h"+X$));               : ' カンジ" インジ"
2090   NEXT J
2100   IF UL=1 THEN LPRINT CHR$(&H1B);"Y";:  ' ESC-Y (アンダ"ーライン カイシ"ョ)
2110   IF KA=1 THEN LPRINT CHR$(&HF);         :' &HF (カクダ"イ カイシ"ョ)
2120   LPRINT CHR$(&H1B);"H";                 : ' ESC-H (カンジ"モード" カイシ"ョ)
2130   SP=0:KA=0:UL=0:KJ=0
2140   RETURN
10000 '
10010 DATA 43,66,21,21,34,56,21,21       :' チュウ カン
10020 DATA 39,4D,21,21,3A,3A,21,21       :' コウ サ
10030 DATA 40,2E,21,21,40,53,21,21       :' セイ セキ
10040 DATA 30,6C,21,21,4D,77,21,21,49,3D: ' イチ ラン ヒョウ
10050 DATA 48,56,39,66                   :' バ"ンゴ"ウ
10060 DATA 3B,61,21,21,21,21,21,21,4C,3E: ' シ メイ
10070 DATA 21,21,39,71,38,6C,3F,74,33,58:' コクゴ" スウガ"ク
10080 DATA 4D,7D,32,4A,3C,52,32,71       : ' リカ シャカイ
10090 DATA 31,51,38,6C,39,67,37,57       :' エイゴ" ゴ"ウケイ
10100 DATA 4A,3F,21,21,36,51             : ' ヘイ キン
```

〔追加後の説明〕

① 行番号10000以降は，帳表に印字する見出しの漢字コードを文字列データとしてDATA文で定義しています。

また，定義している漢字コードが何かをわかりやすくするために，各行の右側にコメントを付けています。(DATA文の後ろにコメントを付ける場合は，必ず「：」で区切る必要があります)

DATA文は，プログラム中のどこにあっても構いません。ここでは，プログラムの一番後ろに置くようにしています。

② 行番号1000～1160は，見出し印字の制御を行うサブルーチンです。

・行番号1020は，見出しの漢字コードデータを最初から読むようにしています。(第2ページ以降の印字のため)

・行番号1030は，改ページを行っています。(&HCは改ページコード)

・行番号1040は，第1見出しの「<u>中　間　考　査　　成　績　一　覧　表</u>」を印字させるものです。(実際の印字は，行番号2010～2140で行います)

・行番号1050の改行命令は，2行以上の改行を行うときに使います。

・行番号1060～1140は，第2見出しの「番号」～「平均」を印字させます。

③ 行番号2010～2140は，見出しを印字するサブルーチンです。

このサブルーチンは，行番号10000～10100のDATA文で定義した見出しの漢字コードを順番に読み込んで印字を行うものです。このサブルーチンを実行するときは，見出し項目を印字する位置までのスペースの数，アンダーラインの有無，拡大文字かどうか，印字する漢字の数などをデータとして次の変数に与えておきます。

ＳＰ………スペースの数

ＵＬ………アンダーラインの有無(ULが1のときアンダーラインを印字)

ＫＡ………拡大文字の設定(KAが1のとき文字を拡大文字にする)

ＫＪ………印字する漢字コードのバイト数

・行番号2020は，変数SPに指定された桁数分のスペースを印字します。

・行番号2040は，変数KAに1が設定されている場合に文字を拡大文字に設定します。拡大文字の解除は，行番号2110で行います。

・行番号2050は，変数ULに1が設定されている場合に印字する文字の下にアンダーラインを引くものです。アンダーラインの解除は，行番号2100で行います。

・行番号2060～2090は，変数KJで指定された数だけDATA文で定義された漢字コードを読み込んで印字しています。漢字は，文字列データとしてDATA文で定義された漢字コードを読み込んで印字しています。漢字コードは，文字列データとしてDATA文で定義されているので，VAL関数を使って数値に変換します。(16進数を表す"&H"を頭に付けます)

設問 3-4 明細を印字する処理ルーチン(＊MEISAI)を追加して下さい。

<追加する処理ルーチン>

```
3000 '
3010 *MEISAI:' <<< ﾒｲｻｲ ｲﾝｼﾞ ｼｮﾘ >>>
3020  GOSUB *DATA.GET
3030  IF SFG$="0" THEN 3110
3040  LPRINT USING "     ##";BG;
3050  LPRINT SPC(3);NM$;
3060  GKO=GKO+KO%:GSU=GSU+SU%:GRI=GRI+RI%
3070  GSHA=GSHA+SHA%:GEI=GEI+EI%:GGO=GGO+GO%
3080  LPRINT " ";USING "  ###";KO%;SU%;RI%;SHA%;EI%;GO%;
3090  LPRINT USING "    ###.#";HEI
3100  LC=LC+1
3110  RETURN
4000 '
4010 *DATA.GET:' <<< ﾃﾞｰﾀ ﾖﾐﾀﾞｼ >>>
4020  GET#1,BG
4030  IF SFG$="0" THEN 4120
4040  NM$=SNM$
4050  KO%=CVI(SKO$)
4060  SU%=CVI(SSU$)
4070  RI%=CVI(SRI$)
4080  SHA%=CVI(SSHA$)
4090  EI%=CVI(SEI$)
4100  GO%=CVI(SGO$)
4110  HEI=CVS(SHE$)
4120  RETURN
```

〔追加後の説明〕

① 成績データの読み込みと各項目の作業用変数へのセットをサブルーチン(＊DATA.GET)で行います。

② 識別コードが「0」のものは，読み飛ばすようにしています。

③ 行番号3040は，出席番号の印字を行います。

④ 行番号3050は，氏名の印字を行います。

⑤ 行番号3060～3070は，各教科の合計を求めています。

⑥ 行番号3080は，各教科の得点と合計点を印字しています。

⑦ 行番号3090は，平均点を印字しています。

⑧ 行番号4020は，成績データの読み込みを行います。

⑨ 行番号4040～4110は，読み込んだデータを作業用の変数へセットしています。このとき，文字型に変換されていた数値はCVI，CVSなどの関数を使って数値型に戻しています。

⑩ 行番号4030は，識別コードが「0」のものを読み飛ばしています。

設問 3-5 クラス平均を印字する処理ルーチン(＊CLASS.HEIKIN)を追加して下さい。

＜追加する処理ルーチン＞

```
5000 '
5010 *CLASS.HEIKIN:' <<< ｸﾗｽﾍｲｷﾝ ｲﾝｼﾞ >>>
5020  LPRINT STRING$(79,"ー")
5030  LPRINT USING "    ##";LC;
5040  SP=0:KJ=4:GOSUB *MID.INJI
5050  SP=3:KJ=20:GOSUB *MID.INJI
5060  LPRINT SPC(5);
5070  LPRINT USING " ###.#";GKO/LC;GSU/LC;GRI/LC;GSHA/LC;
5080  LPRINT USING " ###.#";GEI/LC;GGO/LC;GGO/LC/5
5090  RETURN
10110 DATA 3F,4D,21,21                      : ' ﾆﾝ
10120 DATA 25,2F,21,21,25,69,21,21,25,39: ' ｸ ﾗ ｽ
10130 DATA 21,21,4A,3F,21,21,36,51,21,21: ' ﾍｲ ｷﾝ
```

〔追加後の説明〕

① クラス平均を印字するときに必要な漢字コードの定義と漢字の印字を行います。漢字の印字には，見出し印字ルーチンを利用します。

② 各教科のクラス平均を印字します。

【 プログラムの確認 】

設問 3-6 正しく追加できているかどうかを確認するために，キーインしたプログラムのリストをプリンタに印字して下さい。

＜確認用リスト＞

```
100 ' ｾｲｾｷｲﾁﾗﾝﾋｮｳ ｲﾝｼﾞ      (MONDAI.3)
110 *HAJIME:' <<< ｼｮｷ ｼｮﾘ >>>
120  WIDTH 40,20:CONSOLE ,,0
130  OPEN "2:SEISEK.MST" AS #1
140  FIELD#1,1 AS SFG$,20 AS SNM$,2 AS SKO$
150  FIELD#1,23 AS DM$,2 AS SSU$,2 AS SRI$,2 AS SSHA$
160  FIELD#1,29 AS DM$,2 AS SEI$,2 AS SGO$,4 AS SHE$
165  GOSUB *MIDASHI
170 *MAIN:' <<< ﾒｲﾝ ﾙｰﾁﾝ >>>
180  FOR BG=1 TO LOF(1)
190    GOSUB *MEISAI
200  NEXT BG
220 *OWARI:' <<< ｼｭｳﾘｮｳ ｼｮﾘ >>>
225  GOSUB *CLASS.HEIKIN
230  CLOSE
240  WIDTH 80,25
250  END
1000 '
1010 *MIDASHI:' <<< ﾐﾀﾞｼ ｲﾝｼﾞ ｾｲｷﾞｮ ﾙｰﾁﾝ >>>
1020  RESTORE 10010
1030  LPRINT CHR$(&HC);:LPRINT             : ' &hC (ｶｲﾍﾟｰｼﾞ)
1040  SP=9:KA=1:UL=1:KJ=34:GOSUB *MID.INJI
1050  LPRINT:LPRINT CHR$(&H1F);CHR$(&H10+2);
1060  SP=3 :KJ=4:GOSUB *MID.INJI          : ' SP (ｽﾍﾟｰｽ ﾉ ｶｽﾞ)
1070  SP=3 :KJ=12:GOSUB *MID.INJI         : ' KJ (ｶﾝｼﾞｺｰﾄﾞﾉｶｽﾞ)
1080  SP=15:KJ=4:GOSUB *MID.INJI
1090  SP=3 :KJ=4:GOSUB *MID.INJI
1100  SP=3 :KJ=4:GOSUB *MID.INJI
1110  SP=3 :KJ=4:GOSUB *MID.INJI
1120  SP=3 :KJ=4:GOSUB *MID.INJI
1130  SP=3 :KJ=4:GOSUB *MID.INJI
1140  SP=3 :KJ=6:GOSUB *MID.INJI
1150  LPRINT:LPRINT STRING$(79,"-")
1160  RETURN
2000 '
2010 *MID.INJI:' <<< ﾐﾀﾞｼ ｲﾝｼﾞ ﾙｰﾁﾝ >>>
2020  IF SP><0 THEN LPRINT SPC(SP);
2030  LPRINT CHR$(&H1B);"K";                : ' ESC-K (ｶﾝｼﾞﾓｰﾄﾞ)
2040  IF KA=1 THEN LPRINT CHR$(&HE);        : ' &HE (ｶｸﾀﾞｲ)
2050  IF UL=1 THEN LPRINT CHR$(&H1B);"X";:  : ' ESC-X (ｱﾝﾀﾞｰﾗｲﾝ)
2060  FOR J=1 TO KJ
2070    READ X$                             : ' ｶﾝｼﾞｺｰﾄﾞ ﾖﾐｺﾐ
2080    LPRINT CHR$(VAL("&h"+X$));          : ' ｶﾝｼﾞ ｲﾝｼﾞ
2090  NEXT J
2100  IF UL=1 THEN LPRINT CHR$(&H1B);"Y";:  : ' ESC-Y (ｱﾝﾀﾞｰﾗｲﾝ ｶｲｼﾞｮ)
2110  IF KA=1 THEN LPRINT CHR$(&HF);        : ' &HF (ｶｸﾀﾞｲ ｶｲｼﾞｮ)
2120  LPRINT CHR$(&H1B);"H";                : ' ESC-H (ｶﾝｼﾞﾓｰﾄﾞ ｶｲｼﾞｮ)
2130  SP=0:KA=0:UL=0:KJ=0
2140  RETURN
```

(つづく)

```
3000 '
3010 *MEISAI:' <<< ﾒｲｻｲ ｲﾝｼﾞ ｼｮﾘ >>>
3020  GOSUB *DATA.GET
3030  IF SFG$="0" THEN 3110
3040  LPRINT USING "      ##";BG;
3050  LPRINT SPC(3);NM$;
3060  GKO=GKO+KO%:GSU=GSU+SU%:GRI=GRI+RI%
3070  GSHA=GSHA+SHA%:GEI=GEI+EI%:GGO=GGO+GO%
3080  LPRINT " ";USING "    ###";KO%;SU%;RI%;SHA%;EI%;GO%;
3090  LPRINT USING " ###.#";HEI
3100  LC=LC+1
3110  RETURN
4000 '
4010 *DATA.GET:' <<< ﾃﾞｰﾀ ﾖﾐﾀﾞｼ >>>
4020  GET#1,BG
4030  IF SFG$="0" THEN 4120
4040  NM$=SNM$
4050  KO%=CVI(SKO$)
4060  SU%=CVI(SSU$)
4070  RI%=CVI(SRI$)
4080  SHA%=CVI(SSHA$)
4090  EI%=CVI(SEI$)
4100  GO%=CVI(SGO$)
4110  HEI=CVS(SHE$)
4120  RETURN
5000 '
5010 *CLASS.HEIKIN:' <<< ｸﾗｽﾍｲｷﾝ ｲﾝｼﾞ >>>
5020  LPRINT STRING$(79,"-")
5030  LPRINT USING "  ##";LC;
5040  SP=0:KJ=4:GOSUB *MID.INJI
5050  SP=3:KJ=20:GOSUB *MID.INJI
5060  LPRINT SPC(5);
5070  LPRINT USING " ###.#";GKO/LC;GSU/LC;GRI/LC;GSHA/LC;
5080  LPRINT USING " ###.#";GEI/LC;GGO/LC;GGO/LC/5
5090  RETURN
10000 '
10010 DATA 43,66,21,21,34,56,21,21       : ' ﾁｭｳ ｶﾝ
10020 DATA 39,4D,21,21,3A,3A,21,21       : ' ｺｳ ｻ
10030 DATA 40,2E,21,21,40,53,21,21       : ' ｾｲ ｾｷ
10040 DATA 30,6C,21,21,4D,77,21,21,49,3D: ' ｲﾁ ﾗﾝ ﾋｮｳ
10050 DATA 48,56,39,66                   :'ﾊﾞﾝｺﾞｳ
10060 DATA 3B,61,21,21,21,21,21,21,4C,3E: ' ｼ ﾒｲ
10070 DATA 21,21,39,71,38,6C,3F,74,33,58: '  ｺｸｺﾞ ｽｳｶﾞｸ
10080 DATA 4D,7D,32,4A,3C,52,32,71       : ' ﾘｶ ｼｬｶｲ
10090 DATA 31,51,38,6C,39,67,37,57       : ' ｴｲｺﾞ ｺﾞｳｹｲ
10100 DATA 4A,3F,21,21,36,51             : ' ﾍｲ ｷﾝ
10110 DATA 3F,4D,21,21                   : ' ﾆﾝ
10120 DATA 25,2F,21,21,25,69,21,21,25,39: ' ｸ ﾗ ｽ
10130 DATA 21,21,4A,3F,21,21,36,51,21,21: ' ﾍｲ ｷﾝ
```

【 プログラムの実行 】

設問 3-7 正しく追加できていたら，プログラムを実行して下さい。

＜印字結果例＞

中　間　考　査　成　績　一　覧　表

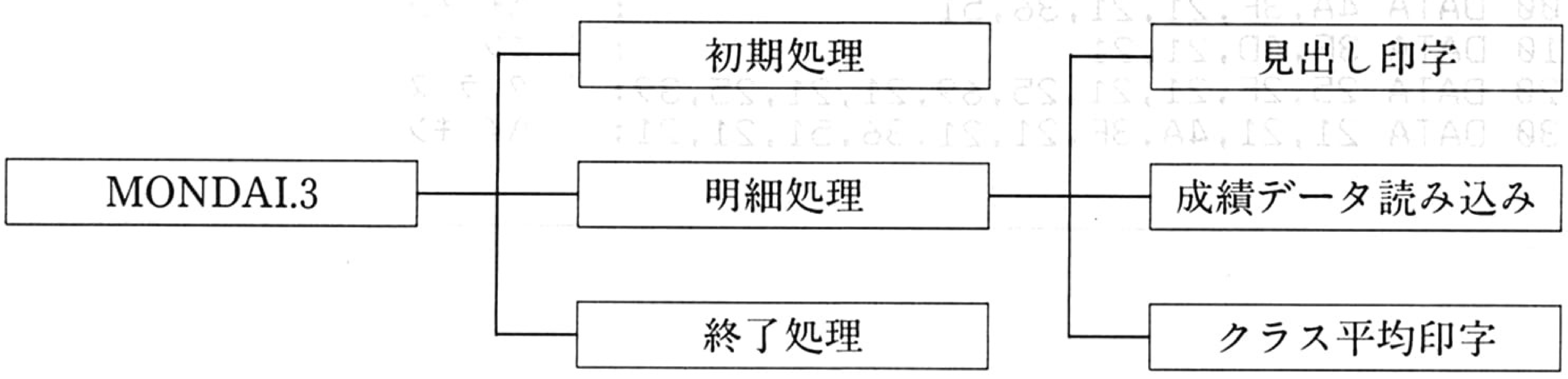
番号	氏　　名	国語	数学	理科	社会	英語	合計	平　均
1	ﾀｶﾊｼ ﾀﾂﾋｺ	87	54	20	54	95	310	62.0
3	ﾀﾅｶ ﾕﾀｶ	60	80	50	50	80	320	64.0
5	ﾊﾗﾀﾞ ｲﾁﾛｳ	90	89	75	54	100	408	81.6
7	ﾔﾏﾀﾞ ﾀﾛｳ	80	91	75	88	95	429	85.8
9	ﾜﾀﾅﾍﾞ ｼﾞﾛｳ	75	75	68	98	58	374	74.8
31	ｶﾜｲ ﾖｼｴ	60	85	95	77	84	401	80.2
33	ｻﾄｳ ｼﾞｭﾝｺ	40	69	45	45	45	244	48.8
35	ｽｽﾞｷ ﾋﾛｺ	20	64	100	65	54	303	60.6
37	ﾏﾂｼﾀ ﾉﾘｺ	64	66	57	84	84	355	71.0
39	ﾓﾘｶﾜ ｱｲｺ	39	45	98	100	54	336	67.2
10人	ク　ラ　ス　平　均	61.5	71.8	68.3	71.5	74.9	348.0	69.6

設問 3-8 印字した結果が正しければ，プログラムをドライブ 2 のフロッピーディスクへ「MONDAI.3」という名前で保存して下さい。

＜実行結果例＞

```
save "2:MONDAI.3"
Ok
```

※　ここで作成したプログラムの構造を次に示します。



【 命令のまとめ 】

① LOF関数

ランダムファイルに登録されているデータのうちで一番大きなレコード番号が何番であるかを調べる関数について説明します。

<一般形式>

LOF(ファイル番号)

・ランダムファイルの中の一番大きなレコード番号を調べます。

・ファイル番号は，一番大きなレコード番号を調べようとするファイルに対してOPEN命令で指定したファイル番号と同じ番号を指定します。

② GET#命令

ランダムファイルからデータを読むための命令について，その一般形式と書き方例を次に示します。

<一般形式>

GET#ファイル番号，レコード番号

・ランダムファイルの中のレコード番号で指定した位置の内容を，FIELD#命令で定義した各項目へ読み込みます。

・ファイル番号は，OPEN命令で指定したファイル番号と同じ番号を指定します。

・レコード番号は，ファイルの中の何件目に書かれているデータを読むのかを指定します。

・レコード番号は，省略することができます。この場合，直前に読み込み(または書き込み)を行った位置の次の位置のデータを読み込みます。なお，プログラムの一番初めにレコード番号を省略してGET#命令を実行するとレコード番号1のデータが読み込まれます。

<書き方例>

① GET#1，5

これは，ファイル番号1のランダムファイルの5番目のレコードを読み込みます。

② GET#1，I

これは，ファイル番号1のランダムファイルのI番目のレコードを読み込みます。

1.3.6 成績データの修正方法 （ランダムファイルの乱読み込みと修正）

前節で成績一覧表を印字しました。この結果によって，成績管理ファイルに登録されているデータのチェックを行うことができます。

ここでは，登録してある成績データに間違いがあったときの，修正方法について学習します。

問題 4 成績データの修正を行うプログラムを作って，次のようにデータを修正して下さい。

出席番号	変更する項目	変更データ
3	国語	92
9	英語	63
31	削　　除	
35	社会	70
39	国語	40

【説　明】

シーケンシャルファイルでは，ファイルをオープンするときに「モード指定」があって，「INPUTモード」なら入力だけ，「OUTPUTモード」なら出力だけ，「APPENDモード」なら追加だけといったように，モード指定によってファイルの処理が限定されました。ところが，ランダムファイルのオープンには，モード指定がありません。これは，ランダムファイルはデータの読み込み，書き込み，追加が自由に行えるからです。

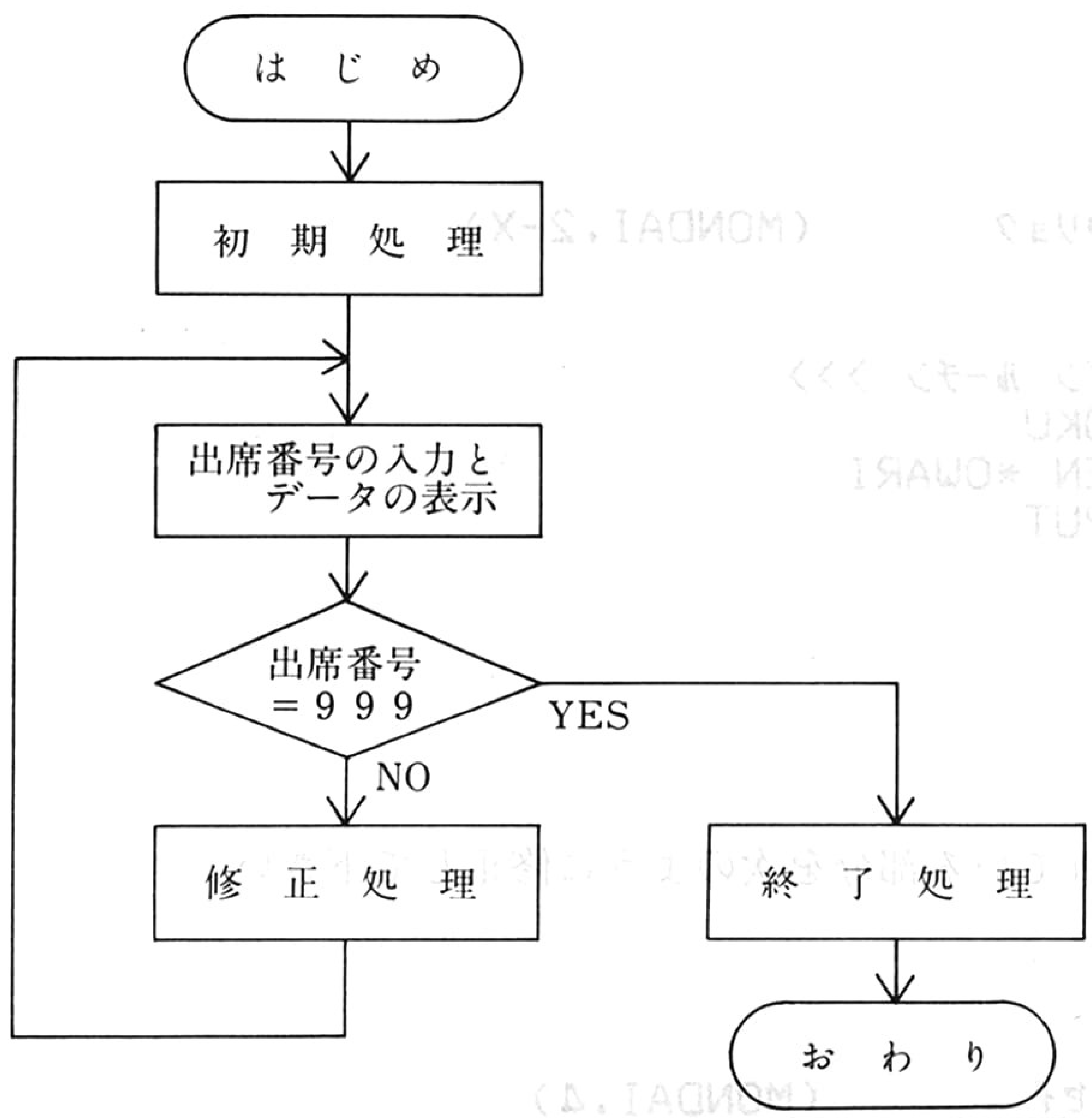
それでは，成績データの修正を行うためのプログラムの処理概要とフローチャートを次に示します。なお，修正を行うときの画面は「1.3.3　成績データの登録方法」で使用した画面を利用します。

<処理概要>

・修正を行う生徒の出席番号を入力してその成績データを修正します。

・出席番号として「999」が入力されたら，処理を終了します。

＜成績データを修正するプログラムのフローチャート＞



【 プログラムの作成 】

ここで作成するプログラムは，出席番号(レコード番号)を入力してそのデータを読み込み，修正を行った後書き込みを行うものです。これは，これまで学習してきたランダムファイルに対する処理をすべて使うことになります。

設問 4-1 ドライブ 2 のフロッピーディスクから「MONDAI.2-X」というプログラムをメモリーにロードして下さい。

＜実行結果例＞

```
load "2:MONDAI.2-X"
Ok
```

〔ロード後の説明〕

① ロードしたプログラムは，ランダムファイル(成績管理ファイル)にデータ(成績データ)を登録するものです。

② このプログラムをもとにして，ランダムファイルのデータ修正を行うプログラムを作成します。

設問 4-2 ロードしたプログラムの行番号100とメインルーチン（行番号170～210）を表示して下さい。

<表示例>

```
list 100
100 ' ｾｲｾｷﾃﾞｰﾀ ﾆｭｳﾘｮｸ        (MONDAI.2-X)
Ok
list 170-210
170 *MAIN:' <<< ﾒｲﾝ ﾙｰﾁﾝ >>>
180  GOSUB *NYURYOKU
190  IF BG=999 THEN *OWARI
200  GOSUB *DATA.PUT
210  GOTO *MAIN
Ok
```

設問 4-3 今画面に表示されている部分を次のように修正して下さい。

```
100 ' ｾｲｾｷﾃﾞｰﾀ ｼｭｳｾｲ        (MONDAI.4)
170 *MAIN:' <<< ﾒｲﾝ ﾙｰﾁﾝ >>>
180  GOSUB *GAMEN.HYOUJI
190  IF BG=999 THEN *OWARI
200  GOSUB *SYUSEI
205  GOSUB *DATA.PUT ────→追加
210  GOTO *MAIN
```

〔修正後の説明〕

① メインルーチンを，成績データの修正を行うように変更しました。

② ＊GAMEN. HYOUJIは，成績データの表示を行うサブルーチンです。

③ ＊SYUSEIは，成績データの修正を行うサブルーチンです。

④ ＊DATA. PUTは，修正された成績データを成績管理ファイルへ書き込むサブルーチンです。

設問 4-4 プログラムの行番号1000～1180を表示して下さい。

<プログラムリスト>

```
list 1000-1180
1000 '
1010 *NYURYOKU:' <<< ﾃﾞｰﾀﾆｭｳﾘｮｸ ｼｮﾘ >>>
1020  CLS:LOCATE  0, 0:PRINT "ｾ ｲ ｾ ｷ ﾃﾞ - ﾀ ﾉ ﾆ ｭ ｳ ﾘ ｮ ｸ"
1030  LOCATE  0, 2:INPUT "1. ﾊﾞﾝｺﾞｳ(999-->ｵﾜﾘ) ";BG
1040  IF BG=999 THEN 1180
1050  IF BG<1 OR BG>50 THEN *NYURYOKU
```

（つづく）

```
1060  LOCATE  0, 4:INPUT "2. ｼﾒｲ      ";NM$
1070  LOCATE  0, 6:INPUT "3. ｺｸｺﾞ     ";KO%
1080  LOCATE  0, 8:INPUT "4. ｽｳｶﾞｸ    ";SU%
1090  LOCATE  0,10:INPUT "5. ﾘｶ      ";RI%
1100  LOCATE  0,12:INPUT "6. ｼｬｶｲ     ";SHA%
1110  LOCATE  0,14:INPUT "7. ｴｲｺﾞ     ";EI%
1120  GO%=KO%+SU%+RI%+SHA%+EI%:HEI=GO%/5
1130  LOCATE  0,15:PRINT STRING$(40,"-")
1140  LOCATE  3,16:PRINT USING "ｺﾞｳｹｲ  = ###";GO%
1150  LOCATE 18,16:PRINT USING "---> ﾍｲｷﾝ = ###.#";HEI
1160  LOCATE  0,18:INPUT "ﾆｭﾘｮｸﾊ ｺﾚﾃﾞｲｲﾃﾞｽｶ (y/n) ";YN$
1170  IF YN$><"Y" AND YN$><"y" THEN *NYURYOKU
1180  RETURN
Ok
```

〔表示後の説明〕

① 表示した部分は，成績データをキーボードから入力するサブルーチンです。

② この部分を利用して，出席番号を入力してその成績データを表示するサブルーチン(*GAMEN.HYOUJI)を作成します。

設問 4-5 表示されているサブルーチンを次のように修正して下さい。

〔修正要領の説明〕

① 既にメモリーに入っているプログラムの命令と同じものをキーインする場合，行番号だけを変更して ⏎ キーを押すことによりコピーすることができました。

ここでは，行番号1000～1180までの命令を利用して行番号260～410に新たな命令を作成します。「1.3.4」の設問2-8では，行番号の付け替えを行って同じ命令をコピーしただけでしたが，ここではコピーする命令に修正を行います。

```
1000 ──→ 260  '
1010 ──→ 270  *GAMEN.HYOUJI:' <<< ｶﾞﾒﾝ ﾋｮｳｼﾞ >>>
1020 ──→ 280   CLS:LOCATE 0, 0:PRINT "ｾ ｲ ｾ ｷ ﾃﾞ ｰ ﾀ ﾉ ﾆ ｭ ｳ ﾘ ｮ ｸ "
1030 ──→ 290   LOCATE 0, 2:INPUT "1. ﾊﾞﾝｺﾞｳ(999-->ｵﾜﾘ) ";BG
1040 ──→ 300   IF BG=999 THEN 410
1050 ──→ 310   IF BG<1 OR BG>50 THEN BEEP:GOTO *GAMEN.HYOUJI
追加 ──→ { 315   GOSUB *DATA.GET
          { 317   IF SFG$="0" THEN BEEP:GOTO *GAMEN.HYOUJI
1060 ──→ 320   LOCATE  0, 4:PRINT "2. ｼﾒｲ      ";NM$
1070 ──→ 330   LOCATE  0, 6:PRINT "3. ｺｸｺﾞ     ";KO%
1080 ──→ 340   LOCATE  0, 8:PRINT "4. ｽｳｶﾞｸ    ";SU%
1090 ──→ 350   LOCATE  0,10:PRINT "5. ﾘｶ      ";RI%
1100 ──→ 360   LOCATE  0,12:PRINT "6. ｼｬｶｲ     ";SHA%
1110 ──→ 370   LOCATE  0,14:PRINT "7. ｴｲｺﾞ     ";EI%
1120 ──────→ 修正しない
1130 ──→ 380   LOCATE  0,15:PRINT STRING$(40,"-")
1140 ──→ 390   LOCATE  3,16:PRINT USING "ｺﾞｳｹｲ  = ###";GO%
1150 ──→ 400   LOCATE 18,16:PRINT USING "---> ﾍｲｷﾝ = ###.#";HEI
1160 }
1170 } ──────→ 修正しない
1180 ──→ 410   RETURN
```

〔修正後の説明〕

① すべて行番号を修正しています。したがって，行番号1000～1180は前のままで，新しく行番号260～410の部分が追加されたことになります。

設問 4-6 次の命令を追加して下さい。

＜追加する命令＞

```
420 '
430 *DATA.GET:' <<< ﾃﾞｰﾀ ﾖﾐﾀﾞｼ >>>
440  GET#1,BG
450  NM$=SNM$
460  KO%=CVI(SKO$)
470  SU%=CVI(SSU$)
480  RI%=CVI(SRI$)
490  SHA%=CVI(SSHA$)
500  EI%=CVI(SEI$)
510  GO%=CVI(SGO$)
520  HEI=CVS(SHE$)
530  RETURN
```

〔追加後の説明〕

① 今追加したサブルーチンは，入力した出席番号の成績データを読み込みそのデータをプログラムの作業用変数にセットするものです。

設問 4-7 行番号1000～1180を表示し直して次のように修正して下さい。

```
1000 '
1010 *SYUSEI:' <<< ｼｭｳｾｲ ｼｮﾘ >>>
1020  LOCATE 0,18:INPUT "ｼｭｳｾｲ ﾊﾞﾝｺﾞｳ (0-->ｻｸｼﾞｮ) ";NO
1030  IF NO<0 OR NO>7 THEN *SYUSEI
1040  IF NO=0 THEN LSET SFG$="0":GOTO 1058
1050  ON NO GOSUB *N1,*N2,*N3,*N4,*N5,*N6,*N7
1060 *N2:LOCATE 0, 4:INPUT "2. ｼﾒｲ   ";NM$:RETURN
1070 *N3:LOCATE 0, 6:INPUT "3. ｺｸｺﾞ  ";KO%:RETURN
1080 *N4:LOCATE 0, 8:INPUT "4. ｽｳｶﾞｸ ";SU%:RETURN
1090 *N5:LOCATE 0,10:INPUT "5. ﾘｶ   ";RI%:RETURN
1100 *N6:LOCATE 0,12:INPUT "6. ｼｬｶｲ  ";SHA%:RETURN
1110 *N7:LOCATE 0,14:INPUT "7. ｴｲｺﾞ  ";EI%:RETURN
1052 *N1:GO%=KO%+SU%+RI%+SHA%+EI%:HEI=GO%/5
1130  LOCATE  0,15:PRINT STRING$(40,"-")          ──→修正しない
1154     LOCATE  3,16:PRINT USING "ｺﾞｳｹｲ   = ###";GO%
1156     LOCATE 18,16:PRINT USING "---> ﾍｲｷﾝ = ###.#";HEI
1160  LOCATE  0,18:INPUT "ﾆｭｳﾘｮｸﾊ ｺﾚﾃﾞｲｲﾃﾞｽｶ (y/n) ";yn$  ┐
1170 IF YN$><"Y" AND YN$><"y" THEN *NYURYOKU           ┘──→修正しない
1058     IF NO<2 THEN RETURN ELSE GOTO 1020

DELETE 1120-1180
```

〔修正後の説明〕

① 修正によって作成したサブルーチンは，画面に表示してある成績データを修正するものです。

② 行番号1020は，修正する項目の番号を入力します。

③ 行番号1040は，修正項目として「0」が入力されたときにその成績データを削除するものとして識別コードに「0」をセットします。

④ 行番号1050は，入力した修正項目の番号に従ってその項目の修正データを入力するようにしています。

修正項目として，「1」(出席番号)が入力されたら，今表示されている生徒の成績データの修正は終わったものとしてその合計と平均を表示して次の処理(出席番号の入力)へ行きます。

設問 4-8 行番号2020を次のように修正して下さい。

＜修正＞

```
2020  IF SFG$="0" THEN 4110
```

〔修正後の説明〕

① 識別コードが「０」の場合(成績データの削除)は，データのセットを行わずに識別コードの書き込みだけを行うようにする。

設問 4-9 行番号を次のように付け直した後，プログラムのリストをプリンタに印字して正しくキーインできているかどうか確認して下さい。

＜行番号の付け直し＞

```
RENUM 1000,260
RENUM 2000,1180
RENUM 3000,2120
RENUM 4000,3160
```

＜確認用リスト＞

```
100 ' ｾｲｾｷﾃﾞｰﾀ ｼｭｳｾｲ        (MONDAI.4)
110 *HAJIME:' <<< ｼｮｷ ｼｮﾘ >>>
120  WIDTH 40,20:CONSOLE ,,0
130  OPEN "2:SEISEK.MST" AS #1
140  FIELD#1,1 AS SFG$,20 AS SNM$,2 AS SKO$
150  FIELD#1,23 AS DM$,2 AS SSU$,2 AS SRI$,2 AS SSHA$
160  FIELD#1,29 AS DM$,2 AS SEI$,2 AS SGO$,4 AS SHE$
170 *MAIN:' <<< ﾒｲﾝ ﾙｰﾁﾝ >>>
180  GOSUB *GAMEN.HYOUJI
190  IF BG=999 THEN *OWARI
200  GOSUB *SYUSEI
205  GOSUB *DATA.PUT
210  GOTO *MAIN
220 *OWARI:' <<< ｼｭｳﾘｮｳ ｼｮﾘ >>>
230  CLOSE
240  WIDTH 80,25
250  END
1000 '
1010 *GAMEN.HYOUJI:' <<< ｶﾞﾒﾝ ﾋｮｳｼﾞ >>>
1020  CLS:LOCATE 0, 0:PRINT "ｾ ｲ ｾ ｷ ﾃﾞ ｰ ﾀ ﾉ ﾆ ｭ ｳ ﾘ ｮ ｸ"
1030  LOCATE 0, 2:INPUT "1. ﾊﾞﾝｺﾞｳ(999-->ｵﾜﾘ) ";BG
1040  IF BG=999 THEN 1170
1050  IF BG<1 OR BG>50 THEN BEEP:GOTO *GAMEN.HYOUJI
1060  GOSUB *DATA.GET
1070  IF SFG$="0" THEN BEEP:GOTO *GAMEN.HYOUJI
1080  LOCATE  0, 4:PRINT "2. ｼﾒｲ     ";NM$
1090  LOCATE  0, 6:PRINT "3. ｺｸｺﾞ    ";KO%
1100  LOCATE  0, 8:PRINT "4. ｽｳｶﾞｸ   ";SU%
1110  LOCATE  0,10:PRINT "5. ﾘｶ      ";RI%
1120  LOCATE  0,12:PRINT "6. ｼｬｶｲ    ";SHA%
1130  LOCATE  0,14:PRINT "7. ｴｲｺﾞ    ";EI%
1140  LOCATE  0,15:PRINT STRING$(40,"-")
1150  LOCATE  3,16:PRINT USING "ｺﾞｳｹｲ  = ###";GO%
1160  LOCATE 18,16:PRINT USING "---> ﾍｲｷﾝ = ###.#";HEI
1170  RETURN
2000 '
2010 *DATA.GET:' <<< ﾃﾞｰﾀ ﾖﾐﾀﾞｼ >>>
2020  GET#1,BG
2030  NM$=SNM$
2040  KO%=CVI(SKO$)
2050  SU%=CVI(SSU$)
2060  RI%=CVI(SRI$)
2070  SHA%=CVI(SSHA$)
2080  EI%=CVI(SEI$)
2090  GO%=CVI(SGO$)
2100  HEI=CVS(SHE$)
2110  RETURN
3000 '
3010 *SYUSEI:' <<< ｼｭｳｾｲ ｼｮﾘ >>>
3020  LOCATE  0,18:INPUT "ｼｭｳｾｲ ﾊﾞﾝｺﾞｳ (0-->ｻｸｼﾞｮ) ";NO
3030  IF NO<0 OR NO>7 THEN *SYUSEI
3040  IF NO=0 THEN LSET SFG$="0":GOTO 3090
```

（つづく）

```
3050  ON NO GOSUB *N1,*N2,*N3,*N4,*N5,*N6,*N7
3060 *N1:GO%=KO%+SU%+RI%+SHA%+EI%:HEI=GO%/5
3070     LOCATE  3,16:PRINT USING "ｺﾞｳｹｲ  = ###";GO%
3080     LOCATE 18,16:PRINT USING "---> ﾍｲｷﾝ = ###.#";HEI
3090     IF NO<2 THEN RETURN ELSE GOTO 3020
3100 *N2:LOCATE 0, 4:INPUT "2. ｼﾒｲ     ";NM$:RETURN
3110 *N3:LOCATE 0, 6:INPUT "3. ｺｸｺﾞ    ";KO%:RETURN
3120 *N4:LOCATE 0, 8:INPUT "4. ｽｳｶﾞｸ   ";SU%:RETURN
3130 *N5:LOCATE 0,10:INPUT "5. ﾘｶ      ";RI%:RETURN
3140 *N6:LOCATE 0,12:INPUT "6. ｼｬｶｲ    ";SHA%:RETURN
3150 *N7:LOCATE 0,14:INPUT "7. ｴｲｺﾞ    ";EI%:RETURN
4000 '
4010 *DATA.PUT:' <<< ﾃﾞｰﾀﾉ ｶｷｺﾐ >>>
4020  IF SFG$="0" THEN 4110
4030  LSET SNM$=NM$
4040  LSET SKO$=MKI$(KO%)
4050  LSET SSU$=MKI$(SU%)
4060  LSET SRI$=MKI$(RI%)
4070  LSET SSHA$=MKI$(SHA%)
4080  LSET SEI$=MKI$(EI%)
4090  LSET SGO$=MKI$(GO%)
4100  LSET SHE$=MKS$(HEI)
4110  PUT#1,BG
4120  RETURN
```

設問 4-10　正しくキーインできていたら，プログラムを実行して成績データの修正を行って下さい。

＜修正する成績データ＞

出席番号	変更する項目	変更データ
3	国語	92
9	英語	63
31	削　　除	
35	社会	70
39	国語	40

＜実行例＞

```
ｾ ｲ ｾ ｷ ﾃﾞ ｰ ﾀ ﾉ ﾆ ｭ ｳ ﾘ ｮ ｸ

1. ﾊﾞﾝｺﾞｳ(999-->ｵﾜﾘ) ? 3

2. ｼﾒｲ       ﾀﾅｶ ﾕﾀｶ

3. ｺｸｺﾞ      60

4. ｽｳｶﾞｸ     80

5. ﾘｶ        50

6. ｼｬｶｲ      50

7. ｴｲｺﾞ      80
________________________________________
   ｺﾞｳｹｲ  = 320    ---> ﾍｲｷﾝ =   64.0

ｼｭｳｾｲ ﾊﾞﾝｺﾞｳ (0-->ｻｸｼﾞｮ) ? 3
```

⇩

```
ｾ ｲ ｾ ｷ ﾃﾞ ｰ ﾀ ﾉ ﾆ ｭ ｳ ﾘ ｮ ｸ

1. ﾊﾞﾝｺﾞｳ(999-->ｵﾜﾘ) ? 3

2. ｼﾒｲ       ﾀﾅｶ ﾕﾀｶ

3. ｺｸｺﾞ      ? 92

4. ｽｳｶﾞｸ     80

5. ﾘｶ        50

6. ｼｬｶｲ      50

7. ｴｲｺﾞ      80
________________________________________
   ｺﾞｳｹｲ  = 352    ---> ﾍｲｷﾝ =   70.4

ｼｭｳｾｲ ﾊﾞﾝｺﾞｳ (0-->ｻｸｼﾞｮ) ?
```

※　修正番号を入力するときに，何も入力しないで ⏎ キーだけを押すと「0」とみなされ，その成績データを削除(識別コードを0に)してしまうので注意して下さい。

設問 4-11　プログラムをドライブ 2 のフロッピーディスクへ「MONDAI.4」という名前で保存して下さい。

＜実行結果例＞

```
save "2:MONDAI.4"
Ok
```

・なお，修正した結果は「MONDAI.3」というプログラムを実行して確認します。

<印字結果例>

中　間　考　査　成　績　一　覧　表

番号	氏　名	国語	数学	理科	社会	英語	合計	平 均
1	ﾀｶﾊｼ ﾀﾂﾋｺ	87	54	20	54	95	310	62.0
3	ﾀﾅｶ ﾕﾀｶ	92	80	50	50	80	352	70.4
5	ﾊﾗﾀﾞ ｲﾁﾛｳ	90	89	75	54	100	408	81.6
7	ﾔﾏﾀﾞ ﾀﾛｳ	80	91	75	88	95	429	85.8
9	ﾜﾀﾅﾍﾞ ｼﾞﾛｳ	75	75	68	98	63	379	75.8
33	ｻﾄｳ ｼﾞｭﾝｺ	40	69	45	45	45	244	48.8
35	ｽｽﾞｷ ﾋﾛｺ	20	64	100	70	54	308	61.6
37	ﾏﾂｼﾀ ﾉﾘｺ	64	66	57	84	84	355	71.0
39	ﾓﾘｶﾜ ｱｲｺ	40	45	98	100	54	337	67.4
9人	ク ラ ス 平 均	65.3	70.3	65.3	71.4	74.4	346.9	69.4

ここで作成したプログラム(MONDAI.4)は，成績データの修正を行うものですが，成績データの追加はできません。そこで，成績データの追加ができるようにプログラムを改造することにします。

設問 4-12　次のようにプログラムを修正して下さい。

<修正>

```
100 ' セイセキデ゛ータ シュウセイ          (MONDAI.4-X)
1070 ←────削除
2030  IF SFG$="0" THEN NM$="" ELSE NM$=SNM$
2040  IF SFG$="0" THEN KO%=0  ELSE KO%=CVI(SKO$)
2050  IF SFG$="0" THEN SU%=0  ELSE SU%=CVI(SSU$)
2060  IF SFG$="0" THEN RI%=0  ELSE RI%=CVI(SRI$)
2070  IF SFG$="0" THEN SHA%=0 ELSE SHA%=CVI(SSHA$)
2080  IF SFG$="0" THEN EI%=0  ELSE EI%=CVI(SEI$)
2090  IF SFG$="0" THEN GO%=0  ELSE GO%=CVI(SGO$)
2100  IF SFG$="0" THEN HEI=0  ELSE HEI=CVS(SHE$)
3011  IF SFG$="1" THEN 3020 ELSE LSET SFG$="1"
3012  GOSUB *N2
3013  GOSUB *N3
3014  GOSUB *N4
3015  GOSUB *N5
3016  GOSUB *N6
3017  GOSUB *N7
3018  GOSUB *N1
```

〔修正後の説明〕

① 行番号2030～2100の修正は，識別コードが「0」の場合に各項目の作業用変数に初期値をセットし，それ以外のときは読み込んだデータをセットするものです。

② 行番号3011～3018は，識別コードが「0」のときに各項目をすべて入力するようにしています。

設問 4-13 プログラムのリストをプリンタに印字して確認して下さい。

<確認用リスト>

```
100 ′ ｾｲｾｷﾃﾞｰﾀ ｼｭｳｾｲ          (MONDAI.4-X)
110 *HAJIME:′ <<< ｼｮｷ ｼｮﾘ >>>
120  WIDTH 40,20:CONSOLE ,,0
130  OPEN "2:SEISEK.MST" AS #1
140  FIELD#1,1 AS SFG$,20 AS SNM$,2 AS SKO$
150  FIELD#1,23 AS DM$,2 AS SSU$,2 AS SRI$,2 AS SSHA$
160  FIELD#1,29 AS DM$,2 AS SEI$,2 AS SGO$,4 AS SHE$
170 *MAIN:′ <<< ﾒｲﾝ ﾙｰﾁﾝ >>>
180  GOSUB *GAMEN.HYOUJI
190  IF BG=999 THEN *OWARI
200  GOSUB *SYUSEI
205  GOSUB *DATA.PUT
210  GOTO *MAIN
220 *OWARI:′ <<< ｼｭｳﾘｮｳ ｼｮﾘ >>>
230  CLOSE
240  WIDTH 80,25
250  END
1000 ′
1010 *GAMEN.HYOUJI:′ <<< ｶﾞﾒﾝ ﾋｮｳｼﾞ >>>
1020  CLS:LOCATE 0, 0:PRINT "ｾ ｲ ｾ ｷ ﾃﾞ ｰ ﾀ ﾉ ﾆ ｭ ｳ ﾘ ｮ ｸ "
1030  LOCATE 0, 2:INPUT "1. ﾊﾞﾝｺﾞｳ(999-->ｵﾜﾘ) ";BG
1040  IF BG=999 THEN 1170
1050  IF BG<1 OR BG>50 THEN BEEP:GOTO *GAMEN.HYOUJI
1060  GOSUB *DATA.GET
1080  LOCATE  0, 4:PRINT "2. ｼﾒｲ     ";NM$
1090  LOCATE  0, 6:PRINT "3. ｺｸｺﾞ    ";KO%
1100  LOCATE  0, 8:PRINT "4. ｽｳｶﾞｸ   ";SU%
1110  LOCATE  0,10:PRINT "5. ﾘｶ      ";RI%
1120  LOCATE  0,12:PRINT "6. ｼｬｶｲ    ";SHA%
1130  LOCATE  0,14:PRINT "7. ｴｲｺﾞ    ";EI%
1140  LOCATE  0,15:PRINT STRING$(40,"-")
1150  LOCATE  3,16:PRINT USING "ｺﾞｳｹｲ  = ###";GO%
1160  LOCATE 18,16:PRINT USING "---> ﾍｲｷﾝ = ###.#";HEI
1170  RETURN
2000 ′
2010 *DATA.GET:′ <<< ﾃﾞｰﾀ ﾖﾐﾀﾞｼ >>>
2020  GET#1,BG
2030  IF SFG$="0" THEN NM$="" ELSE NM$=SNM$
2040  IF SFG$="0" THEN KO%=0  ELSE KO%=CVI(SKO$)
```

（つづく）

```
2050  IF SFG$="0" THEN SU%=0  ELSE SU%=CVI(SSU$)
2060  IF SFG$="0" THEN RI%=0  ELSE RI%=CVI(SRI$)
2070  IF SFG$="0" THEN SHA%=0 ELSE SHA%=CVI(SSHA$)
2080  IF SFG$="0" THEN EI%=0  ELSE EI%=CVI(SEI$)
2090  IF SFG$="0" THEN GO%=0  ELSE GO%=CVI(SGO$)
2100  IF SFG$="0" THEN HEI=0  ELSE HEI=CVS(SHE$)
2110  RETURN
3000 '
3010 *SYUSEI:' <<< ｼｭｳｾｲ ｼｮﾘ >>>
3011  IF SFG$="1" THEN 3020 ELSE LSET SFG$="1"
3012  GOSUB *N2
3013  GOSUB *N3
3014  GOSUB *N4
3015  GOSUB *N5
3016  GOSUB *N6
3017  GOSUB *N7
3018  GOSUB *N1
3020  LOCATE  0,18:INPUT "ｼｭｳｾｲ ﾊﾞﾝｺﾞｳ (0-->ｻｸｼﾞｮ) ";NO
3030  IF NO<0 OR NO>7 THEN *SYUSEI
3040  IF NO=0 THEN LSET SFG$="0":GOTO 3090
3050  ON NO GOSUB *N1,*N2,*N3,*N4,*N5,*N6,*N7
3060 *N1:GO%=KO%+SU%+RI%+SHA%+EI%:HEI=GO%/5
3070     LOCATE  3,16:PRINT USING "ｺﾞｳｹｲ  = ###";GO%
3080     LOCATE 18,16:PRINT USING "---> ﾍｲｷﾝ = ###.#";HEI
3090     IF NO<2 THEN RETURN ELSE GOTO 3020
3100 *N2:LOCATE 0, 4:INPUT "2. ｼﾒｲ     ";NM$:RETURN
3110 *N3:LOCATE 0, 6:INPUT "3. ｺｸｺﾞ   ";KO%:RETURN
3120 *N4:LOCATE 0, 8:INPUT "4. ｽｳｶﾞｸ  ";SU%:RETURN
3130 *N5:LOCATE 0,10:INPUT "5. ﾘｶ     ";RI%:RETURN
3140 *N6:LOCATE 0,12:INPUT "6. ｼｬｶｲ   ";SHA%:RETURN
3150 *N7:LOCATE 0,14:INPUT "7. ｴｲｺﾞ   ";EI%:RETURN
4000 '
4010 *DATA.PUT:' <<< ﾃﾞｰﾀﾉ ｶｷｺﾐ >>>
4020  IF SFG$="0" THEN 4110
4030  LSET SNM$=NM$
4040  LSET SKO$=MKI$(KO%)
4050  LSET SSU$=MKI$(SU%)
4060  LSET SRI$=MKI$(RI%)
4070  LSET SSHA$=MKI$(SHA%)
4080  LSET SEI$=MKI$(EI%)
4090  LSET SGO$=MKI$(GO%)
4100  LSET SHE$=MKS$(HEI)
4110  PUT#1,BG
4120  RETURN
```

設問 4-14　プログラムを実行して，次のデータを追加して下さい。

＜追加する成績データ＞

出席番号	氏　名	国語	数学	理科	社会	英語
4	タジツ　ユウイチ	80	70	60	50	62
31	カワイ　ヨシエ	60	85	95	77	84

＜実行例＞

```
セ イ セ キ デ ー タ ノ ニ ュ ウ リ ョ ク

1. バンゴウ(999-->オワリ) ? 4

2. シメイ      ?

3. コクゴ      0

4. スウガク    0

5. リカ        0

6. シャカイ    0

7. エイゴ      0
------------------------------------------------
   ゴウケイ   =   0    ---> ヘイキン  =   0
```

```
セ イ セ キ デ ー タ ノ ニ ュ ウ リ ョ ク

1. バンゴウ(999-->オワリ) ? 4

2. シメイ      ? タジツ ユウイチ

3. コクゴ      ? 80

4. スウガク    ? 70

5. リカ        ? 60

6. シャカイ    ? 50

7. エイゴ      ? 62
------------------------------------------------
   ゴウケイ   = 322    ---> ヘイキン  =  64.4

シュウセイ バンゴウ (0-->サクジョ) ?
```

〔実行後の説明〕

① このプログラムによって，成績データの修正，削除，追加などの更新処理が自由にできます。

② 一度削除したデータは，すべて初期値で表示されます。

設問 4-15　このプログラムをドライブ 2 のフロッピーディスクへ「MONDAI.4-X」という名前で保存して下さい。

＜実行結果例＞

```
save "2:MONDAI.4-X"
Ok
```

・なお，修正した結果は「MONDAI.3」というプログラムを実行して確認します。

＜印字結果例＞

中　間　考　査　成　績　一　覧　表

番号	氏　　名	国語	数学	理科	社会	英語	合計	平　均
1	ﾀｶﾊｼ ﾀﾂﾋｺ	87	54	20	54	95	310	62.0
3	ﾀﾅｶ ﾕﾀｶ	92	80	50	50	80	352	70.4
4	ﾀｼﾞﾂ ﾕｳｲﾁ	80	70	60	50	62	322	64.4
5	ﾊﾗﾀﾞ ｲﾁﾛｳ	90	89	75	54	100	408	81.6
7	ﾔﾏﾀﾞ ﾀﾛｳ	80	91	75	88	95	429	85.8
9	ﾜﾀﾅﾍﾞ ｼﾞﾛｳ	75	75	68	98	63	379	75.8
31	ｶﾜｲ ﾖｼｴ	60	85	95	77	84	401	80.2
33	ｻﾄｳ ｼﾞｭﾝｺ	40	69	45	45	45	244	48.8
35	ｽｽﾞｷ ﾋﾛｺ	20	64	100	70	54	308	61.6
37	ﾏﾂｼﾀ ﾉﾘｺ	64	66	57	84	84	355	71.0
39	ﾓﾘｶﾜ ｱｲｺ	40	45	98	100	54	337	67.4
11人	ク ラ ス 平 均	66.2	71.6	67.5	70.0	74.2	349.5	69.9

1.3.7 氏名を漢字にするには（漢字データの入出力）

これまで作成してきたプログラムは，生徒の名前をカナ文字で扱っていました。ここでは，それを漢字で扱えるようにするため次のプログラムを改造します。

- 成績管理ファイル初期設定プログラム（MONDAI.1）
- 成績管理ファイル更新プログラム　（MONDAI.4-X）
- 成績一覧表印字プログラム　（MONDAI.3）

（1）初期設定プログラム（MONDAI.1）の改造

初期設定プログラムでは，氏名の初期化を普通のスペースでなく漢字のスペース（&H2121）で行うように改造します。

〔改造手順〕

① 初めに「MONDAI.1」のプログラムをロードします。

② 次のリストのようにアンダーライン部分を修正します。

＜リスト＞

```
10 OPEN "2:SEISEK.MST" AS #1
20  FIELD#1,1 AS SFG$,20 AS SNM$,2 AS SKO$,2 AS SSU$,2 AS SRI$,2 AS SSHA$,2 AS S
EI$,2 AS SGO$,4 AS SHE$
30 LSET SFG$="0"
40 LSET SNM$=STRING$(20,&H21)
50 LSET SKO$=MKI$(0)
60 LSET SSU$=MKI$(0)
70 LSET SRI$=MKI$(0)
80 LSET SSHA$=MKI$(0)
90 LSET SEI$=MKI$(0)
100 LSET SGO$=MKI$(0)
110 LSET SHE$=MKS$(0)
120 FOR I=1 TO 50
130   PUT#1,I
140 NEXT I
150 CLOSE
160 END
```

③ このプログラムをドライブ2のフロッピーディスクに「MONDAI.K1」という名前で保存しておきます。

（2）更新プログラム（MONDAI.4-X）の改造

更新プログラムでは，氏名を漢字で扱うために次の二つの部分を改造します。

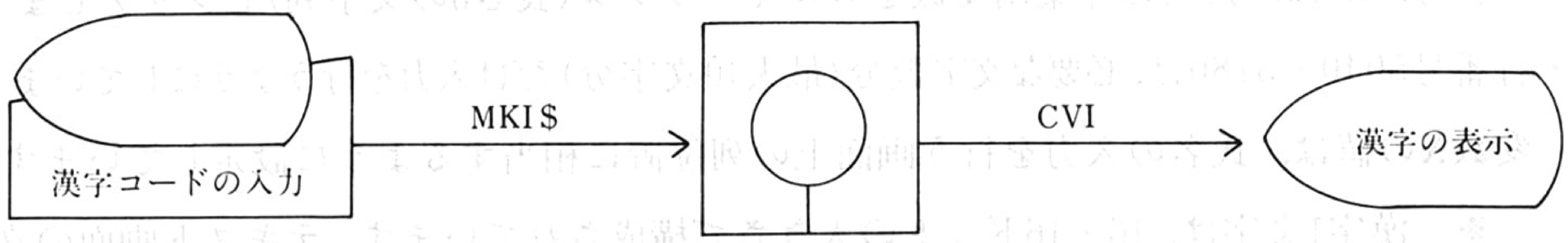
- （i）氏名をキーボードから入力する部分
- （ii）成績データの氏名を画面に表示する部分

（i）の改造では，キーボードから漢字コードを入力してその漢字を画面に表示し，入力が終了したらそれまで入力した漢字コードを作業用変数NM＄（成績データの氏名項目の作業用変数）にセットします。

(ii)の改造では，漢字コードで入っている成績データの氏名を画面に表示するように改造します。

漢字コードをファイルに記録しておく場合には，コード自体を文字データとして記録するよりも，漢字コードが整数型の数値であることに注目してMKI＄関数で変換して記録した方が効率がよくなります。

<漢字コードの入力と記録>



〔改造手順〕

① 初めに「MONDAI.4-X」というプログラムをロードします。

② (ｉ)の改造として次のように修正および追加を行います。

<改造リスト>

```
100 ' ｾｲｾｷﾃﾞｰﾀ ｺｳｼﾝ         (MONDAI.K4)
120  WIDTH 40,20:SCREEN 2:CONSOLE ,,0
240  WIDTH 80,25:CLS 3:SCREEN 0
3100 *N2:LOCATE 0, 4:PRINT "2. ｼﾒｲ     ? ":GOSUB *KANJI.GET:RETURN
5000 '
5010 *KANJI.GET:' <<< ｶﾝｼﾞ ﾆｭｳﾘｮｸ >>>
5020  CLS 2
5030  NM$=""
5040  FOR X=12 TO 21
5050    LOCATE X-1,4:PRINT "  ";
5060    LOCATE X,4:PRINT "■";
5070    KN$=""
5080    A$=INKEY$:IF A$="" THEN 5080
5090    IF A$=CHR$(&H7F) THEN GOSUB *DEL:GOTO 5060
5100    IF A$=CHR$(&HD)  THEN X=X+1:GOTO 5190
5110    IF A$=CHR$(&H20) THEN KN$="2121":GOTO 5140
5120    KN$=KN$+A$
5130    IF LEN(KN$)><4 THEN 5080
5140    KN%=VAL("&h"+KN$)
5150    IF KN%=0 THEN BEEP:GOTO 5060
5160    PUT (X*16,80),KANJI(KN%)
5170    NM$=NM$+MKI$(KN%)
5180  NEXT X
5190  LOCATE X-1,4:PRINT " "
5200  RETURN
6000 '
6010 *DEL:' <<< ｶﾝｼﾞ ﾄﾘｹｼ >>>
6020  IF LEN(KN$)>0 THEN 6080
6030  IF X=12 THEN NM$="":GOTO 6080
6040  X=X-1
6050  NM$=LEFT$(NM$,LEN(NM$)-2)
6060  LOCATE X+1,4:PRINT "  ";
6070  PUT (X*16,80),KANJI(&H2121),PSET
6080  RETURN
```

追加（5000～6080）

<改造後の説明>

・行番号3100は，氏名の入力を漢字入力サブルーチン（＊KANJI.GET）で行うようにしています。

・行番号5000～5200は，漢字入力を行うサブルーチンです。また，行番号6000～6080は，漢字コード入力中にDELキーが押されたときの処理です。

・行番号5020は，漢字を表示する画面(グラフィック画面)をクリアします。

・行番号5030は，氏名の作業用変数をヌルキャラクタ(長さ0の文字列)でクリアします。

・行番号5040～5180は，必要な文字数分(最大10文字分)だけ入力を行うようにしています。変数Xの値は，氏名の入力を行う画面上の列位置に相当するように設定しています。

※　漢字1文字は，16×16ドットの大きさで構成されています。テキスト画面の文字が40×20サイズのとき，その1文字は16×16ドットに相当します。

・行番号5050と5060は，現在の入力位置にカーソルの代わりとしてグラフィック文字の"■"を表示します。

・行番号5070は，1字分の漢字コードを入力する変数KN＄のクリアを行います。

・行番号5080は，漢字コードを一つずつ入力します。

・行番号5090～5110は，入力した文字が特別なもののとき，それに対する処置を行います。

入力が DEL キーのとき，漢字コードの取り消し処理を行います。

入力が ⏎ キーのとき，漢字コードの入力を終了します。

入力がスペースのとき，自動的に漢字のスペースに変換します。

・行番号5080～5130は，一つずつ入力した漢字コードが4文字(漢字1文字分の漢字コード)になるまで入力を繰り返すようにしています。

・行番号5140は，文字データとして入力した漢字コードを整数に変換しています。(漢字コードは，通常16進で表されていますが10進の整数でも構いません)

・行番号5160は，入力した漢字コードに対応する漢字を表示します。

・行番号5170は，入力した漢字コードを1文字分ずつMKI＄関数で文字列に変換した後，変数NM＄に加えて(つなげて)いきます。

・行番号6000～6080の削除処理は，DELキーが押されたとき，1文字分の漢字コードの入力途中であればそこまでの入力をキャンセルし，それ以外のときは前に入力した漢字を削除します。

③ (ii)の改造として，次のように修正および追加を行います。

＜改造リスト＞

```
1020  CLS 3:LOCATE 0, 0:PRINT "ｾ ｲ ｾ ｷ ﾃﾞ ｰ ﾀ ﾉ ﾆ ｭ ｳ ﾘ ｮ ｸ"
1080  LOCATE  0, 4:PRINT "2. ｼﾒｲ     ":GOSUB *KANJI.PUT
2030  IF SFG$="0" THEN NM$=STRING$(20,&H21) ELSE NM$=SNM$
7000 '
7010 *KANJI.PUT:' <<< ｶﾝｼﾞ ｼｭﾂﾘｮｸ >>>
7020  I=1
7030  FOR X=12 TO 21
7040    KN%=CVI(MID$(NM$,I,2))
7050    IF KN%=0 THEN 7090
7060    PUT (X*16,80),KANJI(KN%)
7070    I=I+2
7080  NEXT X
7090  RETURN
```
（7000～7090 → 追加）

＜改造後の説明＞

・行番号1080は，氏名の表示を漢字出力サブルーチン（*KANJI.PUT）で行うようにしています。

・行番号7030～7080は，変数NM＄に文字列として入っている漢字コードを漢字1字分ずつ取り出して，CVI関数で整数に戻した後漢字を表示しています。

④ (i)および(ii)の改造を行った結果プログラムは次のようになります。

＜確認用リスト＞

```
100 ' ｾｲｾｷﾃﾞｰﾀ ｺｳｼﾝ        (MONDAI.K4)
110 *HAJIME:' <<< ｼｮｷ ｼｮﾘ >>>
120  WIDTH 40,20:SCREEN 2:CONSOLE ,,0
130  OPEN "2:SEISEK.MST" AS #1
140  FIELD#1,1 AS SFG$,20 AS SNM$,2 AS SKO$
150  FIELD#1,23 AS DM$,2 AS SSU$,2 AS SRI$,2 AS SSHA$
160  FIELD#1,29 AS DM$,2 AS SEI$,2 AS SGO$,4 AS SHE$
170 *MAIN:' <<< ﾒｲﾝ ﾙｰﾁﾝ >>>
180  GOSUB *GAMEN.HYOUJI
190  IF BG=999 THEN *OWARI
200  GOSUB *SYUSEI
205  GOSUB *DATA.PUT
210  GOTO *MAIN
220 *OWARI:' <<< ｼｭｳﾘｮｳ ｼｮﾘ >>>
230  CLOSE
240  WIDTH 80,25:CLS 3:SCREEN 0
250  END
1000 '
1010 *GAMEN.HYOUJI:' <<< ｶﾞﾒﾝ ﾋｮｳｼﾞ >>>
1020  CLS 3:LOCATE 0, 0:PRINT "ｾ ｲ ｾ ｷ ﾃﾞ ｰ ﾀ ﾉ ﾆ ｭ ｳ ﾘ ｮ ｸ"
1030  LOCATE 0, 2:INPUT "1. ﾊﾞﾝｺﾞｳ(999-->ｵﾜﾘ) ";BG
1040  IF BG=999 THEN 1170
1050  IF BG<1 OR BG>50 THEN BEEP:GOTO *GAMEN.HYOUJI
1060  GOSUB *DATA.GET
1080  LOCATE  0, 4:PRINT "2. ｼﾒｲ     ":GOSUB *KANJI.PUT
1090  LOCATE  0, 6:PRINT "3. ｺｸｺﾞ    ";KO%
1100  LOCATE  0, 8:PRINT "4. ｽｳｶﾞｸ   ";SU%
1110  LOCATE  0,10:PRINT "5. ﾘｶ     ";RI%
1120  LOCATE  0,12:PRINT "6. ｼｬｶｲ    ";SHA%
```
（つづく）

```
1130  LOCATE  0,14:PRINT "7. ｴｲｺﾞ    ";EI%
1140  LOCATE  0,15:PRINT STRING$(40,"-")
1150  LOCATE  3,16:PRINT USING "ｺﾞｳｹｲ  = ###";GO%
1160  LOCATE 18,16:PRINT USING "---> ﾍｲｷﾝ = ###.#";HEI
1170  RETURN
2000 '
2010 *DATA.GET:' <<< ﾃﾞｰﾀ ﾖﾐﾀﾞｼ >>>
2020  GET#1,BG
2030  IF SFG$="0" THEN NM$=STRING$(20,&H21) ELSE NM$=SNM$
2040  IF SFG$="0" THEN KO%=0  ELSE KO%=CVI(SKO$)
2050  IF SFG$="0" THEN SU%=0  ELSE SU%=CVI(SSU$)
2060  IF SFG$="0" THEN RI%=0  ELSE RI%=CVI(SRI$)
2070  IF SFG$="0" THEN SHA%=0 ELSE SHA%=CVI(SSHA$)
2080  IF SFG$="0" THEN EI%=0  ELSE EI%=CVI(SEI$)
2090  IF SFG$="0" THEN GO%=0  ELSE GO%=CVI(SGO$)
2100  IF SFG$="0" THEN HEI=0  ELSE HEI=CVS(SHE$)
2110  RETURN
3000 '
3010 *SYUSEI:' <<< ｼｭｳｾｲ ｼｮﾘ >>>
3011  IF SFG$="1" THEN 3020 ELSE LSET SFG$="1"
3012  GOSUB *N2
3013  GOSUB *N3
3014  GOSUB *N4
3015  GOSUB *N5
3016  GOSUB *N6
3017  GOSUB *N7
3018  GOSUB *N1
3020  LOCATE  0,18:INPUT "ｼｭｳｾｲ ﾊﾞﾝｺﾞｳ (0-->ｻｸｼﾞｮ) ";NO
3030  IF NO<0 OR NO>7 THEN *SYUSEI
3040  IF NO=0 THEN LSET SFG$="0":GOTO 3090
3050  ON NO GOSUB *N1,*N2,*N3,*N4,*N5,*N6,*N7
3060 *N1:GO%=KO%+SU%+RI%+SHA%+EI%:HEI=GO%/5
3070      LOCATE  3,16:PRINT USING "ｺﾞｳｹｲ  = ###";GO%
3080      LOCATE 18,16:PRINT USING "---> ﾍｲｷﾝ = ###.#";HEI
3090      IF NO<2 THEN RETURN ELSE GOTO 3020
3100 *N2:LOCATE 0, 4:PRINT "2. ｼﾒｲ     ? ":GOSUB *KANJI.GET:RETURN
3110 *N3:LOCATE 0, 6:INPUT "3. ｺｸｺﾞ     ";KO%:RETURN
3120 *N4:LOCATE 0, 8:INPUT "4. ｽｳｶﾞｸ    ";SU%:RETURN
3130 *N5:LOCATE 0,10:INPUT "5. ﾘｶ       ";RI%:RETURN
3140 *N6:LOCATE 0,12:INPUT "6. ｼｬｶｲ     ";SHA%:RETURN
3150 *N7:LOCATE 0,14:INPUT "7. ｴｲｺﾞ     ";EI%:RETURN
4000 '
4010 *DATA.PUT:' <<< ﾃﾞｰﾀﾉ ｶｷｺﾐ >>>
4020  IF SFG$="0" THEN 4110
4030  LSET SNM$=NM$
4040  LSET SKO$=MKI$(KO%)
4050  LSET SSU$=MKI$(SU%)
4060  LSET SRI$=MKI$(RI%)
4070  LSET SSHA$=MKI$(SHA%)
4080  LSET SEI$=MKI$(EI%)
4090  LSET SGO$=MKI$(GO%)
4100  LSET SHE$=MKS$(HEI)
4110  PUT#1,BG
4120  RETURN
5000 '
5010 *KANJI.GET:' <<< ｶﾝｼﾞ ﾆｭｳﾘｮｸ >>>
5020  CLS 2
5030  NM$=""
5040  FOR X=12 TO 21
5050    LOCATE X-1,4:PRINT " ";
5060    LOCATE X,4:PRINT "■";
5070    KN$=""
5080    A$=INKEY$:IF A$="" THEN 5080
5090    IF A$=CHR$(&H7F) THEN GOSUB *DEL:GOTO 5060
```

(つづく)

```
5100     IF A$=CHR$(&HD)  THEN X=X+1:GOTO 5190
5110     IF A$=CHR$(&H20) THEN KN$="2121":GOTO 5140
5120     KN$=KN$+A$
5130     IF LEN(KN$)><4 THEN 5080
5140     KN%=VAL("&h"+KN$)
5150     IF KN%=0 THEN BEEP:GOTO 5060
5160     PUT (X*16,80),KANJI(KN%)
5170     NM$=NM$+MKI$(KN%)
5180   NEXT X
5190   LOCATE X-1,4:PRINT "."
5200   RETURN
6000  '
6010  *DEL:' <<< ｶﾝｼﾞ ﾄﾘｹｼ >>>
6020   IF LEN(KN$)>0 THEN 6080
6030   IF X=12 THEN NM$="":GOTO 6080
6040   X=X-1
6050   NM$=LEFT$(NM$,LEN(NM$)-2)
6060   LOCATE X+1,4:PRINT " ";
6070   PUT (X*16,80),KANJI(&H2121),PSET
6080   RETURN
7000  '
7010  *KANJI.PUT:' <<< ｶﾝｼﾞ ｼｭﾂﾘｮｸ >>>
7020   I=1
7030   FOR X=12 TO 21
7040     KN%=CVI(MID$(NM$,I,2))
7050     IF KN%=0 THEN 7090
7060     PUT (X*16,80),KANJI(KN%)
7070     I=I+2
7080   NEXT X
7090   RETURN
```

⑤　このプログラムをドライブ2のフロッピーディスクに「MONDAI.K4」という名前で保存しておきます。

（3）　成績一覧表印字プログラム（MONDAI.3）の改造

成績一覧表印字プログラムでは，氏名を漢字で印字するように改造します。

〔改造手順〕

①　初めに「MONDAI.3」というプログラムをロードします。

②　次のように修正および追加を行います。

```
100 ' ｾｲｾｷｲﾁﾗﾝﾋｮｳ ｲﾝｼﾞ          (MONDAI.K3)
3050  LPRINT SPC(3);:GOSUB *NAMAE.INJI
3080  LPRINT SPC(6);USING "    ###";KO%;SU%;RI%;SHA%;EI%;GO%;
3120 '                                        ┐
3130 *NAMAE.INJI:' <<< ﾅﾏｴﾉ ｶﾝｼﾞｲﾝｼﾞ >>>     │
3140  LPRINT CHR$(&H1B);"K";                  │
3150  FOR J=1 TO 20 STEP 2                    │
3160    X$=MID$(NM$,J,2)                      │
3170    IF X$="" THEN X$=MKI$(&H2121)         ├──→追加
3180    LPRINT RIGHT$(X$,1);LEFT$(X$,1);      │
3190  NEXT J                                  │
3200  LPRINT CHR$(&H1B);"H";                  │
3210  RETURN                                  ┘
```

<改造後の説明>

・漢字をプリンタに印字するときは，通常プリンタを漢字印字モードにした後，漢字コード(2バイト)をCHR＄関数を使って1バイトずつ文字に変換して印字します。ところが，ここでの場合漢字コードはファイルへ登録する際にMKI＄関数ですでに文字に変換されています。したがって，ここでのプリンタへの印字は通常の場合と違ってCHR＄関数で文字に変換する必要はなく，そのまま1バイトずつ印字すればよいことになります。このとき，注意することはMKI＄関数によって変換された文字は次のように入れ替わっているということです。

(例)　「亜(&H3021)」という漢字コードをファイルへ登録する場合

KN%
&H3021

NM$=MKI$ (KN%)

NM＄	
&H21	&H30

入れ替わる

・行番号3050は，氏名の印字を漢字印字サブルーチン(＊NAMAE.INJI)で行うようにしています。

・行番号3140は，プリンタを漢字印字モードにしています。

・行番号3150～3190は，氏名の項目から2バイト(漢字1字分)ずつ取り出して，その順番を入れ替えてプリンタに印字します。

・変数Jは，氏名の項目から2バイトずつ取り出すときの取り出し位置です。

・行番号3200は，プリンタを普通の印字モードにしています。

・改造した結果プログラムは次のようになります。

```
100 ´ ｾｲｾｷｲﾁﾗﾝﾋｮｳ ｲﾝｼﾞ          (MONDAI.K3)
110 *HAJIME:´ <<< ｼｮｷ ｼｮﾘ >>>
120  WIDTH 40,20:CONSOLE ,,0
130  OPEN "2:SEISEK.MST" AS #1
140  FIELD#1,1 AS SFG$,20 AS SNM$,2 AS SKO$
150  FIELD#1,23 AS DM$,2 AS SSU$,2 AS SRI$,2 AS SSHA$
160  FIELD#1,29 AS DM$,2 AS SEI$,2 AS SGO$,4 AS SHE$
165  GOSUB *MIDASHI
170 *MAIN:´ <<< ﾒｲﾝ ﾙｰﾁﾝ >>>
180  FOR BG=1 TO LOF(1)
190    GOSUB *MEISAI
200  NEXT BG
220 *OWARI:´ <<< ｼｭｳﾘｮｳ ｼｮﾘ >>>
225  GOSUB *CLASS.HEIKIN
230  CLOSE
240  WIDTH 80,25
250  END
1000 ´
1010 *MIDASHI:´ <<< ﾐﾀﾞｼ ｲﾝｼﾞ ｾｲｷﾞｮ ﾙｰﾁﾝ >>>
1020  RESTORE 10010
1030  LPRINT CHR$(&HC);:LPRINT           : ´ &hC (ｶｲﾍﾟｰｼﾞ)
1040  SP=9:KA=1:UL=1:KJ=34:GOSUB *MID.INJI
1050  LPRINT:LPRINT CHR$(&H1F);CHR$(&H10+2);
```

(つづく)

```
1060  SP=3 :KJ=4:GOSUB *MID.INJI              : ' SP (スペース ノ カズ)
1070  SP=3 :KJ=12:GOSUB *MID.INJI              : ' KJ (カンジコードノカズ)
1080  SP=15:KJ=4:GOSUB *MID.INJI
1090  SP=3 :KJ=4:GOSUB *MID.INJI
1100  SP=3 :KJ=4:GOSUB *MID.INJI
1110  SP=3 :KJ=4:GOSUB *MID.INJI
1120  SP=3 :KJ=4:GOSUB *MID.INJI
1130  SP=3 :KJ=4:GOSUB *MID.INJI
1140  SP=3 :KJ=6:GOSUB *MID.INJI
1150  LPRINT:LPRINT STRING$(79,"-")
1160  RETURN
2000 '
2010 *MID.INJI:' <<< ミダシ インジ ルーチン >>>
2020  IF SP><0 THEN LPRINT SPC(SP);
2030  LPRINT CHR$(&H1B);"K";               : ' ESC-K (カンジモード)
2040  IF KA=1 THEN LPRINT CHR$(&HE);        : ' &HE (カクダイ)
2050  IF UL=1 THEN LPRINT CHR$(&H1B);"X";:  ' ESC-X (アンダーライン)
2060  FOR J=1 TO KJ
2070    READ X$                              : ' カンジコード ヨミコミ
2080    LPRINT CHR$(VAL("&h"+X$));           : ' カンジ インジ
2090  NEXT J
2100  IF UL=1 THEN LPRINT CHR$(&H1B);"Y";:  ' ESC-Y (アンダーライン カイジョ)
2110  IF KA=1 THEN LPRINT CHR$(&HF);        : ' &HF (カクダイ カイジョ)
2120  LPRINT CHR$(&H1B);"H";               : ' ESC-H (カンジモード カイジョ)
2130  SP=0:KA=0:UL=0:KJ=0
2140  RETURN
3000 '
3010 *MEISAI:' <<< メイサイ インジ ショリ >>>
3020  GOSUB *DATA.GET
3030  IF SFG$="0" THEN 3110
3040  LPRINT USING "    ##";BG;
3050  LPRINT SPC(3);:GOSUB *NAMAE.INJI
3060  GKO=GKO+KO%:GSU=GSU+SU%:GRI=GRI+RI%
3070  GSHA=GSHA+SHA%:GEI=GEI+EI%:GGO=GGO+GO%
3080  LPRINT SPC(6);USING "    ###";KO%;SU%;RI%;SHA%;EI%;GO%;
3090  LPRINT USING "    ###.#";HEI
3100  LC=LC+1
3110  RETURN
3120 '
3130 *NAMAE.INJI:' <<< ナマエノ カンジインジ >>>
3140  LPRINT CHR$(&H1B);"K";
3150  FOR J=1 TO 20 STEP 2
3160    X$=MID$(NM$,J,2)
3170    IF X$="" THEN X$=MKI$(&H2121)
3180    LPRINT RIGHT$(X$,1);LEFT$(X$,1);
3190  NEXT J
3200  LPRINT CHR$(&H1B);"H";
3210  RETURN
4000 '
4010 *DATA.GET:' <<< データ ヨミダシ >>>
4020  GET#1,BG
4030  IF SFG$="0" THEN 4120
4040  NM$=SNM$
4050  KO%=CVI(SKO$)
4060  SU%=CVI(SSU$)
4070  RI%=CVI(SRI$)
4080  SHA%=CVI(SSHA$)
4090  EI%=CVI(SEI$)
4100  GO%=CVI(SGO$)
4110  HEI=CVS(SHE$)
```

(つづく)

```
4120  RETURN
5000 '
5010 *CLASS.HEIKIN:' <<< ｸﾗｽﾍｲｷﾝ ｲﾝｼﾞ >>>
5020  LPRINT STRING$(79,"-")
5030  LPRINT USING "     ##";LC;
5040  SP=0:KJ=4:GOSUB *MID.INJI
5050  SP=3:KJ=20:GOSUB *MID.INJI
5060  LPRINT SPC(5);
5070  LPRINT USING " ###.#";GKO/LC;GSU/LC;GRI/LC;GSHA/LC;
5080  LPRINT USING " ###.#";GEI/LC;GGO/LC;GGO/LC/5
5090  RETURN
10000 '
10010 DATA 43,66,21,21,34,56,21,21       : ' ﾁｭｳ ｶﾝ
10020 DATA 39,4D,21,21,3A,3A,21,21       : ' ｺｳ ｻ
10030 DATA 40,2E,21,21,40,53,21,21       : ' ｾｲ ｾｷ
10040 DATA 30,6C,21,21,4D,77.21.21,49.3D: ' ｲﾁ ﾗﾝ ﾋｮｳ
10050 DATA 48,56,39,66                   : ' ﾊﾞﾝｺﾞｳ
10060 DATA 3B,61,21,21,21,21,21,21,4C,3E: ' ｼ ﾒｲ
10070 DATA 21,21,39,71,38,6C,3F,74,33,58: '  ｺｸｺﾞ ｽｳｶﾞｸ
10080 DATA 4D,7D,32,4A,3C,52,32,71       : ' ﾘｶ ｼｬｶｲ
10090 DATA 31,51,38,6C,39,67,37,57       : ' ｴｲｺﾞ ｺﾞｳｹｲ
10100 DATA 4A,3F,21,21,36,51             : ' ﾍｲ ｷﾝ
10110 DATA 3F,4D,21,21                   : ' ﾆﾝ
10120 DATA 25,2F,21,21,25,69,21,21,25,39: ' ｸ ﾗ ｽ
10130 DATA 21,21,4A,3F,21,21,36,51.21,21: ' ﾍｲ ｷﾝ
```

③　このプログラムをドライブ2のフロッピーディスクへ「MONDAI.K3」という名前で保存しておきます。

【 プログラムの実行 】

改造した3本のプログラムの実行は，今までに行ってきた方法と同じ方法で行います。ただし，氏名の入力は16進数の漢字コードで行います。

<登録する成績データの例>

出席番号	氏　　名	国　語	数　学	理　科	社　会	英　語
1	高橋　達彦	87	54	20	54	95
3	田中　豊	92	80	50	50	80
5	原田　一朗	90	89	75	54	100
7	山田　多郎	80	91	75	88	95
9	渡辺　次郎	75	75	68	98	63
31	河合　由絵	60	85	95	77	84
33	佐藤　純子	40	69	45	45	45
35	鈴木　宏子	20	64	100	70	54
37	松下　紀子	64	66	57	84	84
39	森川　愛子	40	45	98	100	54

＜氏名の漢字コード＞

漢字	コード	漢字	コード	漢字	コード	漢字	コード	漢字	コード
高	3962	田	4544	原	3836	山	3B33	渡	454F
橋	3636	中	4366	田	4544	田	4544	辺	4A55
達	4323	豊	4B2D	一	306C	多	423F	次	3C21
彦	4927			朗	4F2F	郎	4F3A	郎	4F3A

漢字	コード	漢字	コード	漢字	コード	漢字	コード	漢字	コード
河	324F	佐	3A34	鈴	4E6B	松	3E3E	森	3F39
合	3967	藤	4623	木	4C5A	下	323C	川	406E
由	4D33	純	3D63	宏	3928	紀	352A	愛	3026
絵	3328	子	3B52	子	3B52	子	3B52	子	3B52

＜実行例＞

中　間　考　査　成　績　一　覧　表

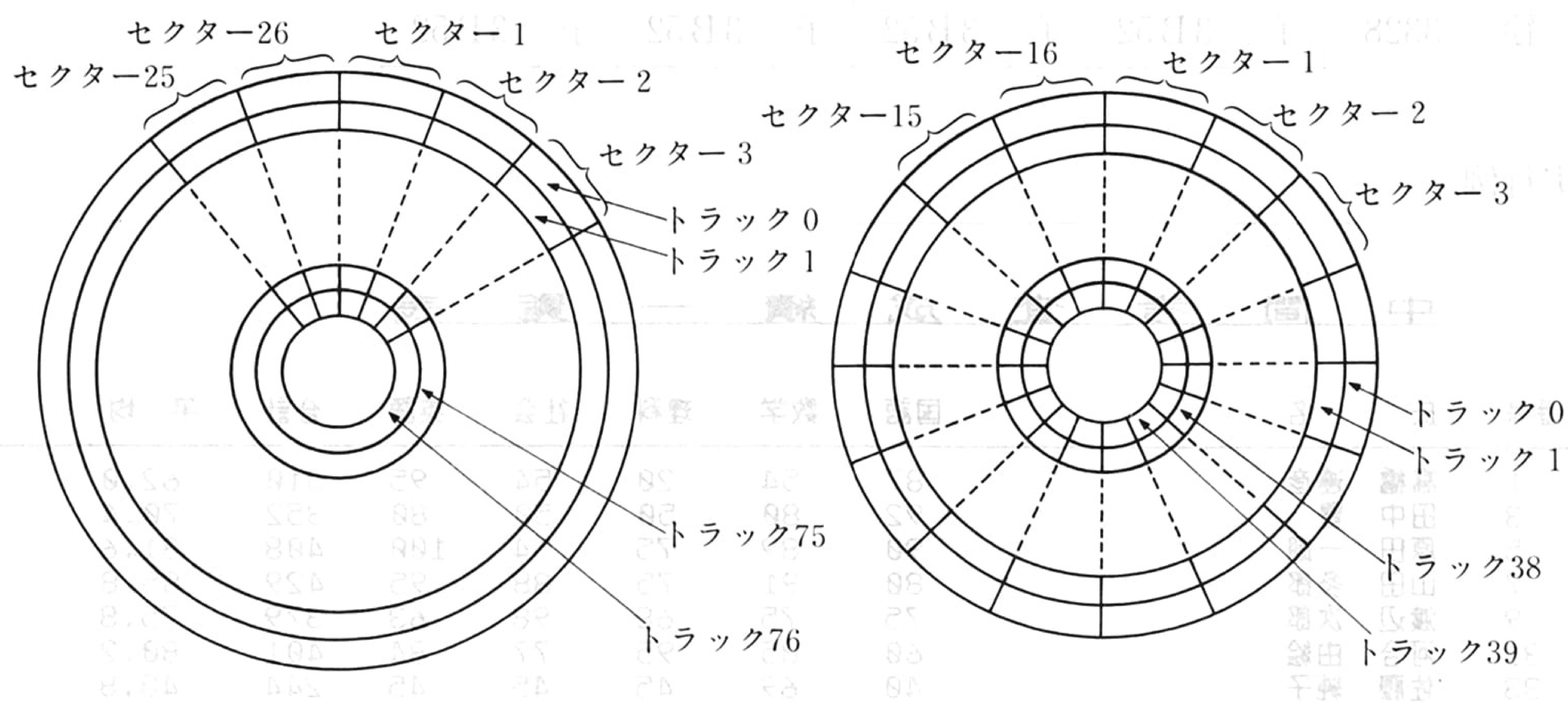
番号	氏　名	国語	数学	理科	社会	英語	合計	平　均
1	高橋　達彦	87	54	20	54	95	310	62.0
3	田中　豊	92	80	50	50	80	352	70.4
5	原田　一朗	90	89	75	54	100	408	81.6
7	山田　多郎	80	91	75	88	95	429	85.8
9	渡辺　次郎	75	75	68	98	63	379	75.8
31	河合　由絵	60	85	95	77	84	401	80.2
33	佐藤　純子	40	69	45	45	45	244	48.8
35	鈴木　宏子	20	64	100	70	54	308	61.6
37	松下　紀子	64	66	57	84	84	355	71.0
39	森川　愛子	40	45	98	100	54	337	67.4
10人	ク ラ ス 平 均	64.8	71.8	68.3	72.0	75.4	352.3	70.5

1.4 ランダムファイルのしくみ

今までは，ランダムファイルの使い方について学習してきました。ここでは，ランダムファイルが実際にどのような形式でフロッピーディスクに記録されるかについて学習します。

1.4.1 フロッピーディスクの構造

ランダムファイルの記録形式を説明する前に，フロッピーディスクの構造について説明します。PC-8801mkIIで使えるフロッピーディスクには，8インチ(直径約20cmの大きさ)の標準フロッピーディスクと5.25インチ(直径約13cmの大きさで一般には5インチと言います)のミニフロッピーディスクとがあります。フロッピーディスクは，次のような理論上の構造に区切られています。



8インチ標準フロッピーディスク　　5インチミニフロッピーディスク

上の図のように1枚のフロッピーディスクは，同心円によって「トラック」というリング状の部分に区切られています。さらに一つのトラックは，いくつかの「セクタ」という部分に分けられていて，それぞれのセクタには「256バイト(256文字)」のデータを記録することができます。

また，PC-8801mkIIが情報の読み書きをする場合は，処理の効率をよくするために，いくつかのセクタをひとまとめにして行っています。この単位を「クラスタ」と呼び，1クラスタの大きさは8インチ標準フロッピーディスクでは26セクタに，5インチミニフロッピーディスクでは8セクタにそれぞれ決められています。

これらの状況をまとめると次のようになります。

フロッピーディスクの種類 ＼ 項目	サーフェス番号	トラック数	トラック番号	セクタ数／トラック	セクタ番号
8インチ(標準)	0(表)，1(裏)	片面当たり77	0〜76	26	1〜26
5インチ(ミニ)	0(表)，1(裏)	片面当たり40	0〜39	16	1〜16

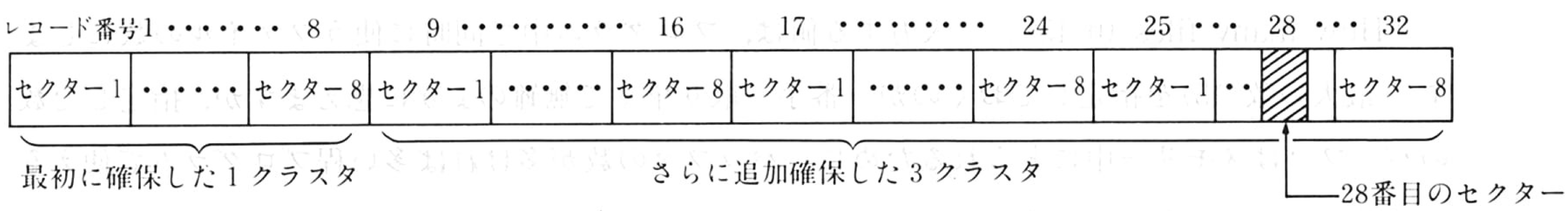
1.4.2　フロッピーディスクとランダムファイルとの関係

ランダムファイルの1レコードは，256桁の固定長です。これは，フロッピーディスクの1セクタに相当します。したがって，ランダムファイルの場合，一般的な使い方をしている限り1レコードと1セクタとは同じ大きさと考えても構いません。

それでは，ランダムファイルがフロッピーディスク上にどのように記録されていくかを説明します。

まず，新しいランダムファイルをオープンしたとき，そのファイルはフロッピーディスク上にありませんから，N_{88}DISK-BASICは，ファイルに必要最低限な領域として1クラスタの領域を用意します。1クラスタは8セクタ(8インチフロッピーの場合は26セクタ)ですから，ランダムファイルの8レコード分が用意されたことになります。

次にデータの書き込みを行うと，そのレコード番号が8以下の場合は今のままで記録されますが，それ以上の場合たとえば28だったとすると，さらに3クラスタの領域が確保され最初に確保したクラスタの先頭セクタから28番目のセクタに記録されることになります。



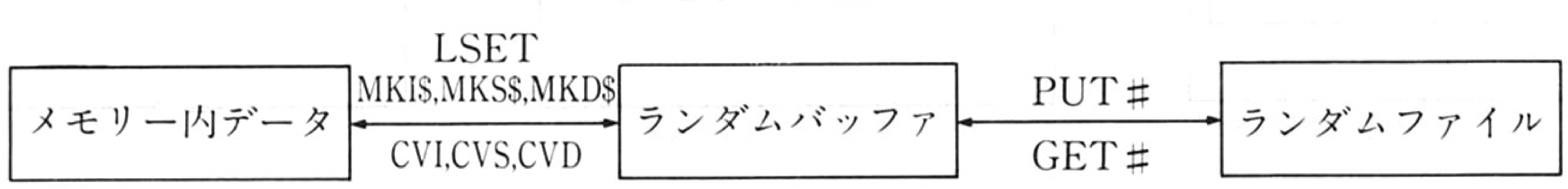
このとき，LOF関数で求められる値(ファイル中の一番大きなレコード番号)は「28」となります。データの読み込みは，「1」～「LOF関数の値」の範囲のレコード番号で行います。　LOF関数の値は，ランダムファイルの大きさを示します。したがって，これ以上の値のレコード番号で(上例では 29以上の値)データを読み込むと「Input past end」のエラーになります。

しかし，LOF関数の値の範囲であれば書き込みを行っていない位置のデータも読み込むことができます。そこで，各レコードに識別コードを持たせてその位置のデータが正しいものであるかどうかを判断する必要が出てくるわけです。(「1.3.2」参照)

LOF関数の値は，それを超えるレコード番号で書き込みが行われたときに更新されます。

1.4.3　ランダムバッファについて

ランダムファイルを利用する場合，FIELD#命令で定義した項目を通してのデータの書き込みや読み込みを行います。これは，メモリーとフロッピーディスクの間でデータをやりとりする「ランダムバッファ」と呼ばれるデータの受け渡し場所があるからです。FIELD#命令の項目定義は，このランダムバッファに対して各項目の長さや配置を定義するものです。



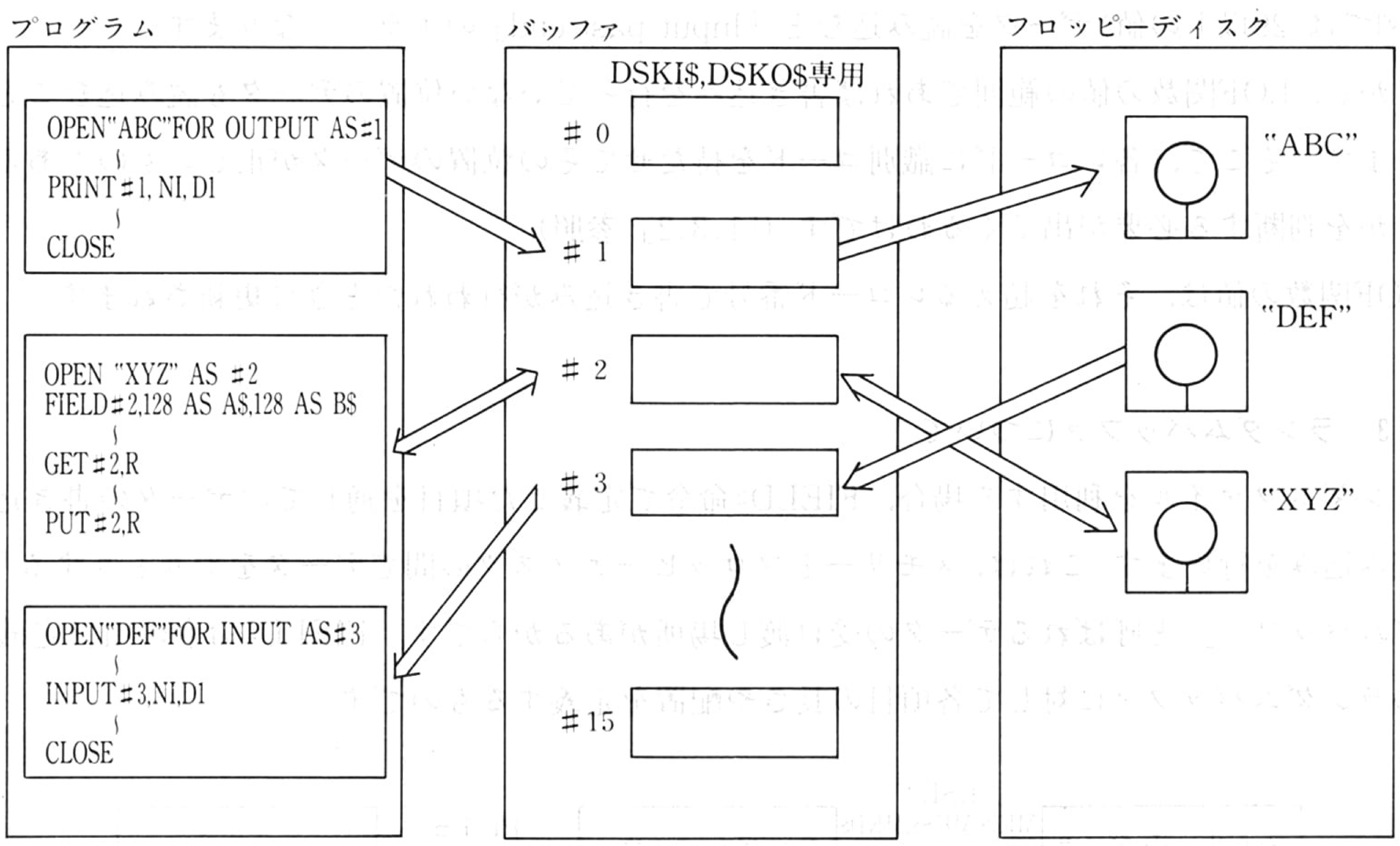
ランダムバッファとは，ランダムファイルで使用するデータの受け渡し場所のことですが，ランダムファイルに限らず，メモリーとフロッピーディスクの間でデータのやり取りを行う場合はその受け渡し場所(バッファ)が必要になります。シーケンシャルファイルを利用するときも，その受け渡し場所はバッファ(入出力バッファともいう)を経由して行われます。しかし，シーケンシャルファイルの場合，ランダムファイルのようにFIELD#命令でレコードの項目をバッファに対して定義する必要はありません。

バッファは，0番～15番までの16個がありますが，PC-8801mkIIを立ち上げるときにその数を変えることができます。(「How many files (0-15) ?」に対して，入力した数がバッファの数になります) フロッピーディスクのファイルを使う場合，必ずファイルのオープンを行いますが，このとき指定するファイル番号はどのバッファを使うかの指定になります。

一つのファイルには必ず一つのバッファを割り当てます。したがって，プログラム中で使うファイルの数だけバッファを用意しておかなければなりません。ただし，一度あるファイルに割り当てたバッファでも，そのファイルをクローズすると他のファイルへ割り当てることができます。

「How many files (0-15)」で入力する値は，プログラム中で同時に使うファイルの数にします。 最大の数(15)を指定しておくのが一番手っ取り早くて無難のように思えますが，指定した数のバッファはメモリー中にとられるために，バッファの数が多ければ多い程プログラムで使えるメモリーが減ってしまいます。

※ バッファ0は，通常は使用しません。これは，DSKI＄という関数とDSKO＄という命令を使って直接フロッピーディスクとデータの受け渡しを行う場合に使用します。



1.5 ランダムファイルの無駄をなくすには

ランダムファイルのレコードは，256桁の固定長で扱われます。したがって，ここまでの学習で習ってきた成績管理ファイルのように1レコード37バイトしか使っていなくても，実際には256バイト(1セクタ)の大きさで書かれてしまいます。このままでは，50件のデータ(正味1850バイト)を登録すると，ランダムファイルの容量として50レコード(12800バイト)も使います。

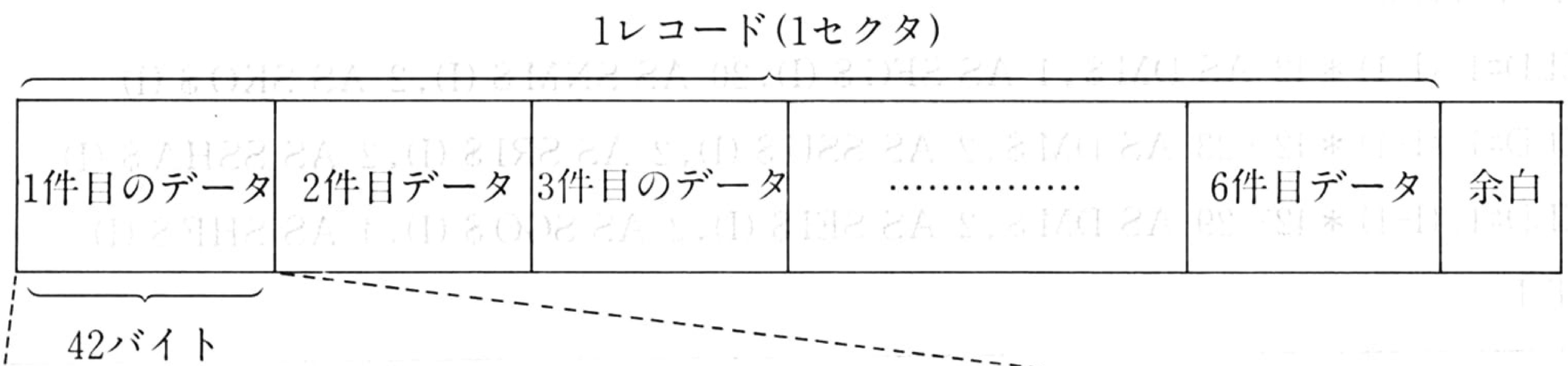
そこで,1レコードに複数件の成績データを格納して成績管理ファイル(ランダムファイル)の有効利用をします。

1.5.1 レコードを効率よく使う

1レコードに複数件のデータを格納することによって,無駄な領域を減らして有効的な使いかたをすることができます。

成績管理ファイルを有効利用したときの1レコード内の項目と順序を次に示します。

・成績管理ファイル (SEISEK.MST)



識別コード(文字)	氏名(文字)	国語(数字)	数学(数字)	理科(数字)	社会(数字)	英語(数字)	合計(数字)	平均(数字)	予備(文字)
1桁	20桁	2桁	2桁	2桁	2桁	2桁	2桁	4桁	5桁
SFG$(1)	SNM$(1)	SKO$(1)	SSU$(1)	SRI$(1)	SSHA$(1)	SEI$(1)	SGO$(1)	SHE$(1)	

※　上の構造の成績データを1レコードに6件分格納します。

今までの項目に加えて，予備(5バイト)の項目を追加しました。

これによって，50件の成績データは9レコード(2304バイト)に納めることができます。

1.5.2 FIELD#命令の定義方法の工夫

1レコードに複数件のデータを格納する場合の項目の定義は,配列変数を使って次のように行います。



```
FIELD#1,1 AS SFG$(1),20 AS SNM$(1),2 AS SKO$(1)
FIELD#1,23 AS DM$,2 AS SSU$(1),2 AS SRI$(1),2 AS SSHA$(1)     } 1件目の
FIELD#1,29 AS DM$,2 AS SEI$(1),2 AS SGO$(1),4 AS SHE$(1)      } データの定義
                    :
                    :
FIELD#1,210 AS DM$,1 AS SFG$(6),20 AS SNM$(6),2 AS SKO$(6)
FIELD#1,233 AS DM$,2 AS SSU$(6),2 AS SRI$(6),2 AS SSHA$(6)    } 6件目の
FIELD#1,239 AS DM$,2 AS SEI$(6),2 AS SGO$(6),4 AS SHE$(6)     } データの定義
```

この定義方法では，FIELD#命令の定義が18行になってしまいます。そこで，FOR～NEXT命令を使って次のように定義します。

```
FOR I=1 TO 6
  FIELD#1,(I-1)*42 AS DM$,1 AS SFG$(I),20 AS SNM$(I),2 AS SKO$(I)
  FIELD#1,(I-1)*42+23 AS DM$,2 AS SSU$(I),2 AS SRI$(I),2 AS SSHA$(I)
  FIELD#1,(I-1)*42+29 AS DM$,2 AS SEI$(I),2 AS SGO$(I),4 AS SHE$(I)
NEXT I
```

1.5.3 レコード番号と出席番号の対応

1レコードに1件のデータを記録していたときは，出席番号が直接レコード番号を表していましたが，複数件のデータを1レコードに納めたために，出席番号とレコード番号が対応しなくなります。

そこで，出席番号からレコード番号とそのレコード内の位置を求める計算式を次に示します。

【 計算式 】

・ レコード番号 ＝ 出席番号 ￥ 6 ＋ 1

　　　　　　　　　　　　　　　└→ 1レコードに含まれるデータ件数

・ 1レコード内のデータ格納位置 ＝ 出席番号 MOD 6

＊ ただし，出席番号が6の倍数(出席番号 MOD 6 が0)のときは，次のようになります。

レコード番号 ＝ 出席番号 ￥ 6

1レコード内のデータ格納位置 ＝ 6

1.5.4 無駄をなくしたプログラム例

1レコードに複数件のデータを格納して無駄をなくしたプログラム例を次に示します。(「1.3.7」で作成したプログラム3本を例に示します)

(1) 初期設定プログラム(MONDAI.K1)の改造例

```
10 OPEN "2:SEISEK.MST" AS #1
20 FOR I=1 TO 6
22   FIELD#1,(I-1)*42 AS DM$,1 AS SFG$(I),20 AS SNM$(I),2 AS SKO$(I)
24   FIELD#1,(I-1)*42+23 AS DM$,2 AS SSU$(I),2 AS SRI$(I),2 AS SSHA$(I)
26   FIELD#1,(I-1)*42+29 AS DM$,2 AS SEI$(I),2 AS SGO$(I),4 AS SHE$(I)
28 NEXT I
29 FOR I=1 TO 6
30   LSET SFG$(I)="0"
40   LSET SNM$(I)=STRING$(20,&H21)
50   LSET SKO$(I)=MKI$(0)
60   LSET SSU$(I)=MKI$(0)
70   LSET SRI$(I)=MKI$(0)
80   LSET SSHA$(I)=MKI$(0)
90   LSET SEI$(I)=MKI$(0)
100  LSET SGO$(I)=MKI$(0)
110  LSET SHE$(I)=MKS$(0)
115 NEXT I
120 FOR I=1 TO 9
130   PUT#1,I
140 NEXT I
150 CLOSE
160 END
```

(2) 更新プログラム(MONDAI.K4)の改造例

```
100 ' ｾｲｾｷﾃﾞｰﾀ ｺｳｼﾝ          (MONDAI.K4X)
110 *HAJIME:' <<< ｼｮｷ ｼｮﾘ >>>
120  WIDTH 40,20:CLS:CONSOLE 19,1
130  OPEN "2:SEISEK.MST" AS #1
135  FOR I=1 TO 6
140    FIELD#1,(I-1)*42 AS DM$,1 AS SFG$(I),20 AS SNM$(I),2 AS SKO$(I)
150    FIELD#1,(I-1)*42+23 AS DM$,2 AS SSU$(I),2 AS SRI$(I),2 AS SSHA$(I)
160    FIELD#1,(I-1)*42+29 AS DM$,2 AS SEI$(I),2 AS SGO$(I),4 AS SHE$(I)
162  NEXT I
170 *MAIN:' <<< ﾒｲﾝ ﾙｰﾁﾝ >>>
180  GOSUB *GAMEN.HYOUJI
190  IF BG=999 THEN *OWARI
200  GOSUB *SYUSEI
205  GOSUB *DATA.PUT
210  GOTO *MAIN
220 *OWARI:' <<< ｼｭｳﾘｮｳ ｼｮﾘ >>>
230  CLOSE
240  WIDTH 80,25
250  END
1000 '
1010 *GAMEN.HYOUJI:' <<< ｶﾞﾒﾝ ﾋｮｳｼﾞ >>>
1020  CONSOLE 0,20:CLS:CONSOLE 19,1
1030  LOCATE 0, 0:PRINT "ｾ ｲ ｾ ｷ ﾃﾞ ｰ ﾀ ﾉ ﾆ ｭ ｳ ﾘ ｮ ｸ "
1040  LOCATE 0, 2:INPUT "1. ﾊﾞﾝｺﾞｳ(999-->ｵﾜﾘ) ";BG
1050  IF BG=999 THEN 1180
1060  IF BG<1 OR BG>50 THEN BEEP:GOTO *GAMEN.HYOUJI
1070  GOSUB *DATA.GET
```

(つづく)

```
1090  LOCATE 0, 4:PRINT "2. ｼﾒｲ      ":GOSUB *KANJI.PUT
1100  LOCATE  0, 6:PRINT "3. ｺｸｺﾞ    ";KO%
1110  LOCATE  0, 8:PRINT "4. ｽｳｶﾞｸ   ";SU%
1120  LOCATE  0,10:PRINT "5. ﾘｶ      ";RI%
1130  LOCATE  0,12:PRINT "6. ｼｬｶｲ    ";SHA%
1140  LOCATE  0,14:PRINT "7. ｴｲｺﾞ    ";EI%
1150  LOCATE  0,15:PRINT STRING$(40,"-")
1160  LOCATE  3,16:PRINT USING "ｺﾞｳｹｲ  = ###";GO%
1170  LOCATE 18,16:PRINT USING "---> ﾍｲｷﾝ = ###.#";HEI
1180  RETURN
2000 '
2010 *DATA.GET:' <<< ﾃﾞｰﾀ ﾖﾐﾀﾞｼ >>>
2012  RNO=BG ¥ 6
2014  DNO=BG MOD 6
2016  IF DNO=0 THEN DNO=6 ELSE RNO=RNO+1
2020  GET#1,RNO
2030  IF SFG$(DNO)="0" THEN NM$="" ELSE NM$=SNM$(DNO)
2040  IF SFG$(DNO)="0" THEN KO%=0  ELSE KO%=CVI(SKO$(DNO))
2050  IF SFG$(DNO)="0" THEN SU%=0  ELSE SU%=CVI(SSU$(DNO))
2060  IF SFG$(DNO)="0" THEN RI%=0  ELSE RI%=CVI(SRI$(DNO))
2070  IF SFG$(DNO)="0" THEN SHA%=0 ELSE SHA%=CVI(SSHA$(DNO))
2080  IF SFG$(DNO)="0" THEN EI%=0  ELSE EI%=CVI(SEI$(DNO))
2090  IF SFG$(DNO)="0" THEN GO%=0  ELSE GO%=CVI(SGO$(DNO))
2100  IF SFG$(DNO)="0" THEN HEI=0  ELSE HEI=CVS(SHE$(DNO))
2110  RETURN
3000 '
3010 *SYUSEI:' <<< ｼｭｳｾｲ ｼｮﾘ >>>
3011  IF SFG$(DNO)="1" THEN 3020 ELSE LSET SFG$(DNO)="1"
3012  GOSUB *N2
3013  GOSUB *N3
3014  GOSUB *N4
3015  GOSUB *N5
3016  GOSUB *N6
3017  GOSUB *N7
3018  GOSUB *N1
3020  LOCATE  0,18:INPUT "ｼｭｳｾｲ ﾊﾞﾝｺﾞｳ (0--->ｻｸｼﾞｮ) ";NO
3030  IF NO<0 OR NO>7 THEN *SYUSEI
3040  IF NO=0 THEN LSET SFG$(DNO)="0":GOTO 3090
3050  ON NO GOSUB *N1,*N2,*N3,*N4,*N5,*N6,*N7
3060 *N1:GO%=KO%+SU%+RI%+SHA%+EI%:HEI=GO%/5
3070      LOCATE  3,16:PRINT USING "ｺﾞｳｹｲ  = ###";GO%
3080      LOCATE 18,16:PRINT USING "---> ﾍｲｷﾝ = ###.#";HEI
3090      IF NO<2 THEN RETURN ELSE GOTO 3020
3100 *N2:LOCATE 0, 4:PRINT "2. ｼﾒｲ     ? ":GOSUB *KANJI.GET:RETURN
3110 *N3:LOCATE 0, 6:INPUT "3. ｺｸｺﾞ    ";KO%:RETURN
3120 *N4:LOCATE 0, 8:INPUT "4. ｽｳｶﾞｸ   ";SU%:RETURN
3130 *N5:LOCATE 0,10:INPUT "5. ﾘｶ      ";RI%:RETURN
3140 *N6:LOCATE 0,12:INPUT "6. ｼｬｶｲ    ";SHA%:RETURN
3150 *N7:LOCATE 0,14:INPUT "7. ｴｲｺﾞ    ";EI%:RETURN
4000 '
4010 *DATA.PUT:' <<< ﾃﾞｰﾀﾉ ｶｷｺﾐ >>>
4020  IF SFG$(DNO)="0" THEN 4110
4030  LSET SNM$(DNO)=NM$
4040  LSET SKO$(DNO)=MKI$(KO%)
4050  LSET SSU$(DNO)=MKI$(SU%)
4060  LSET SRI$(DNO)=MKI$(RI%)
4070  LSET SSHA$(DNO)=MKI$(SHA%)
4080  LSET SEI$(DNO)=MKI$(EI%)
4090  LSET SGO$(DNO)=MKI$(GO%)
4100  LSET SHE$(DNO)=MKS$(HEI)
4110  PUT#1,RNO
4120  RETURN
5000 '
5010 *KANJI.GET:' <<< ｶﾝｼﾞ ﾆｭｳﾘｮｸ >>>
5020  CLS 2
5030  NM$=""
5040  FOR X=12 TO 21
```

(つづく)

```
5050    LOCATE X-1,4:PRINT " ";
5060    LOCATE X,4:PRINT "■";
5070    KN$=""
5080    A$=INKEY$:IF A$="" THEN 5080
5090    IF A$=CHR$(&H7F) THEN GOSUB *DEL:GOTO 5060
5100    IF A$=CHR$(&HD)  THEN X=X+1:GOTO 5190
5110    IF A$=CHR$(&H20) THEN KN$="2121":GOTO 5140
5120    KN$=KN$+A$
5130    IF LEN(KN$)><4 THEN 5080
5140    KN%=VAL("&h"+KN$)
5150    IF KN%=0 THEN BEEP:GOTO 5060
5160    PUT (X*16,80),KANJI(KN%)
5170    NM$=NM$+MKI$(KN%)
5180  NEXT X
5190  LOCATE X-1,4:PRINT " "
5200  RETURN
6000 '
6010 *DEL:' <<< ｶﾝｼﾞ ﾄﾘｹｼ >>>
6020  IF LEN(KN$)>0 THEN 6080
6030  IF X=12 THEN NM$="":GOTO 6080
6040  X=X-1
6050  NM$=LEFT$(NM$,LEN(NM$)-2)
6060  LOCATE X+1,4:PRINT " ";
6070  PUT (X*16,80),KANJI(&H2121),PSET
6080  RETURN
7000 '
7010 *KANJI.PUT:' <<< ｶﾝｼﾞ ｼｭﾂﾘｮｸ >>>
7020  CLS 2
7030  I=1
7040  FOR X=12 TO 21
7050    KN%=CVI(MID$(NM$,I,2))
7060    IF KN%=0 THEN 7100
7070    PUT (X*16,80),KANJI(KN%)
7080    I=I+2
7090  NEXT X
7100  RETURN
```

（3）成績一覧表印字プログラム(MONDAI.K3)の改造例

```
100 ' ｾｲｾｷｲﾁﾗﾝﾋｮｳ ｲﾝｼﾞ          (MONDAI.K3X)
110 *HAJIME:' <<< ｼｮｷ ｼｮﾘ >>>
120  WIDTH 40,20:CLS:CONSOLE 19,1
130  OPEN "2:SEISEK.MST" AS #1
135  FOR I=1 TO 6
140    FIELD#1,(I-1)*42 AS DM$,1 AS SFG$(I),20 AS SNM$(I),2 AS SKO$(I)
150    FIELD#1,(I-1)*42+23 AS DM$,2 AS SSU$(I),2 AS SRI$(I),2 AS SSHA$(I)
160    FIELD#1,(I-1)*42+29 AS DM$,2 AS SEI$(I),2 AS SGO$(I),4 AS SHE$(I)
162  NEXT I
165  GOSUB *MIDASHI
170 *MAIN:' <<< ﾒｲﾝ ﾙｰﾁﾝ >>>
180  FOR BG=1 TO LOF(1)
185    FOR I=1 TO 6
190      GOSUB *MEISAI
200  NEXT I,BG
220 *OWARI:' <<< ｼｭｳﾘｮｳ ｼｮﾘ >>>
225  GOSUB *CLASS.HEIKIN
230  CLOSE
240  WIDTH 80,25
250  END
1000 '
1010 *MIDASHI:' <<< ﾐﾀﾞｼ ｲﾝｼﾞ ｾｲｷﾞｮ ﾙｰﾁﾝ >>>
1020  RESTORE 10010
1030  LPRINT CHR$(&HC);:LPRINT                : ' &hC (ｶｲﾍﾟｰｼﾞ)
1040  SP=9:BG=1:UL=1:KJ=34:GOSUB *MID.INJI
1050  LPRINT:LPRINT CHR$(&H1F);CHR$(&H10+2);
1060  SP=3 :KJ=4:GOSUB *MID.INJI            : ' SP (ｽﾍﾟｰｽ ﾉ ｶｽﾞ)
1070  SP=3 :KJ=12:GOSUB *MID.INJI            : ' KJ (ｶﾝｼﾞｺｰﾄﾞﾉｶｽﾞ)
1080  SP=15:KJ=4:GOSUB *MID.INJI
1090  SP=3 :KJ=4:GOSUB *MID.INJI
1100  SP=3 :KJ=4:GOSUB *MID.INJI
1110  SP=3 :KJ=4:GOSUB *MID.INJI
1120  SP=3 :KJ=4:GOSUB *MID.INJI
1130  SP=3 :KJ=4:GOSUB *MID.INJI
1140  SP=3 :KJ=6:GOSUB *MID.INJI
1150  LPRINT:LPRINT STRING$(79,"-")
1160  RETURN
2000 '
2010 *MID.INJI:' <<< ﾐﾀﾞｼ ｲﾝｼﾞ ﾙｰﾁﾝ >>>
2020  IF SP><0 THEN LPRINT SPC(SP);
2030  LPRINT CHR$(&H1B);"K";                 : ' ESC-K (ｶﾝｼﾞﾓｰﾄﾞ)
2040  IF BG=1 THEN LPRINT CHR$(&HE);         : ' &HE (ｶｸﾀﾞｲ)
2050  IF UL=1 THEN LPRINT CHR$(&H1B);"X";:   ' ESC-X (ｱﾝﾀﾞｰﾗｲﾝ)
2060  FOR J=1 TO KJ
2070    READ X$                               : ' ｶﾝｼﾞｺｰﾄﾞ ﾖﾐｺﾐ
2080    LPRINT CHR$(VAL("&h"+X$));            : ' ｶﾝｼﾞ ｲﾝｼﾞ
2090  NEXT J
2100  IF UL=1 THEN LPRINT CHR$(&H1B);"Y";:   ' ESC-Y (ｱﾝﾀﾞｰﾗｲﾝ ｶｲｼﾞｮ)
2110  IF BG=1 THEN LPRINT CHR$(&HF);         : ' &HF (ｶｸﾀﾞｲ ｶｲｼﾞｮ)
2120  LPRINT CHR$(&H1B);"H";                 : ' ESC-H (ｶﾝｼﾞﾓｰﾄﾞ ｶｲｼﾞｮ)
2130  SP=0:BG=0:UL=0:KJ=0
2140  RETURN
3000 '
3010 *MEISAI:' <<< ﾒｲｻｲ ｲﾝｼﾞ ｼｮﾘ >>>
3020  GOSUB *DATA.GET
3030  IF SFG$(DNO)="0" THEN 3110
3040  LPRINT USING "     ##";BG;
3050  LPRINT SPC(3);:GOSUB *NAMAE.INJI
3060  GKO=GKO+KO%:GSU=GSU+SU%:GRI=GRI+RI%
3070  GSHA=GSHA+SHA%:GEI=GEI+EI%:GGO=GGO+GO%
3080  LPRINT " ";USING "    ###";KO%;SU%;RI%;SHA%;EI%;GO%;
3090  LPRINT USING "    ###.#";HEI
3100  LC=LC+1
3110  RETURN
3120 '
```

（つづく）

```
3130 *NAMAE.INJI:´ <<< ﾅﾏｴﾉ ｶﾝｼﾞｲﾝｼﾞ >>>
3140  LPRINT CHR$(&H1B);"K";
3150  FOR J=1 TO 20 STEP 2
3160    X$=MID$(NM$,J,2)
3170    IF X$="" THEN X$=MKI$(&H2121)
3180    LPRINT RIGHT$(X$,1);LEFT$(X$,1);
3190  NEXT J
3200  LPRINT CHR$(&H1B);"H";
3210  RETURN
4000 ´
4010 *DATA.GET:´ <<< ﾃﾞｰﾀ ﾖﾐﾀﾞｼ >>>
4012  RNO=BG ¥ 6
4014  DNO=BG MOD 6
4016  IF DNO=0 THEN DNO=6 ELSE RNO=RNO+1
4020  IF DNO=1 THEN GET#1,RNO
4030  IF SFG$(DNO)="0" THEN 4120
4040  NM$=SNM$(DNO)
4050  KO%=CVI(SKO$(DNO))
4060  SU%=CVI(SSU$(DNO))
4070  RI%=CVI(SRI$(DNO))
4080  SHA%=CVI(SSHA$(DNO))
4090  EI%=CVI(SEI$(DNO))
4100  GO%=CVI(SGO$(DNO))
4110  HEI=CVS(SHE$(DNO))
4120  RETURN
5000 ´
5010 *CLASS.HEIKIN:´ <<< ｸﾗｽﾍｲｷﾝ ｲﾝｼﾞ >>>
5020  LPRINT STRING$(79,"-")
5030  LPRINT USING "    ##";LC;
5040  SP=0:KJ=4:GOSUB *MID.INJI
5050  SP=3:KJ=20:GOSUB *MID.INJI
5060  LPRINT SPC(5);
5070  LPRINT USING " ###.#";GKO/LC;GSU/LC;GRI/LC;GSHA/LC;
5080  LPRINT USING " ###.#";GEI/LC;GGO/LC;GGO/LC/5
5090  RETURN
10000 ´
10010 DATA 43,66,21,21,34,56,21,21       : ´ ﾁｭｳ ｶﾝ
10020 DATA 39,4D,21,21,3A,3A,21,21       : ´ ｺｳ ｻ
10030 DATA 40,2E,21,21,40,53,21,21       : ´ ｾｲ ｾｷ
10040 DATA 30,6C,21,21,4D,77,21,21,49,3D: ´ ｲﾁ ﾗﾝ ﾋｮｳ
10050 DATA 48,56,39,66                   : ´ ﾊﾞﾝｺﾞｳ
10060 DATA 3B,61,21,21,21,21,21,21,4C,3E: ´ ｼ ﾒｲ
10070 DATA 21,21,39,71,38,6C,3F,74,33,58: ´  ｺｸｺﾞ ｽｳｶﾞｸ
10080 DATA 4D,7D,32,4A,3C,52,32,71       : ´ ﾘｶ ｼｬｶｲ
10090 DATA 31,51,38,6C,39,67,37,57       : ´ ｴｲｺﾞ ｺﾞｳｹｲ
10100 DATA 4A,3F,21,21,36,51             : ´ ﾍｲ ｷﾝ
10110 DATA 3F,4D,21,21                   : ´ ﾆﾝ
10120 DATA 25,2F,21,21,25,69,21,21,25,39: ´ ｸ ﾗ ｽ
10130 DATA 21,21,4A,3F,21,21,36,51,21,21: ´ ﾍｲ ｷﾝ
```

(注) これらのプログラムを実行するときには先に「KILL "SEISEK.MST"」(「3.1.4 KILL」を参照) を実行してから行って下さい。

1.6　プログラム例(2) ——住所録プログラム

ランダムファイルは，どの位置のデータでも読み書きできるために大変便利なものですが，そのためには各データの位置を示すレコード番号をおのおののデータについて把握しておく必要があります。

前例では，出席番号をレコード番号として利用できたので，レコード番号と各データとの結びつきは容易にわかりました。

ここでは，ランダムファイルのプログラム例として住所録プログラムを取り上げます。住所録のデータ項目としては，「氏名」，「住所」，「電話番号」などが考えられます。このようなデータでは，レコード番号として利用できる適当なものもなく，また住所録として利用するにはデータの読み書きを「氏名」で区別して行いたいものです。そこで，索引という考え方をここでは学習することにします。

1.6.1　住所録プログラムの処理と概要

ここで行う住所録管理は，次に示す五つのプログラムで構成されています。

・住所録ファイルの初期化

住所録データファイル(ランダムファイル)の初期設定と，索引用のファイル(シーケンシャルファイル)の確保を行います。

・住所録データの更新

住所録データの登録，削除，修正を行います。

・ソート

住所録データを氏名で五十音順にソートします。

・住所録の一覧表印字

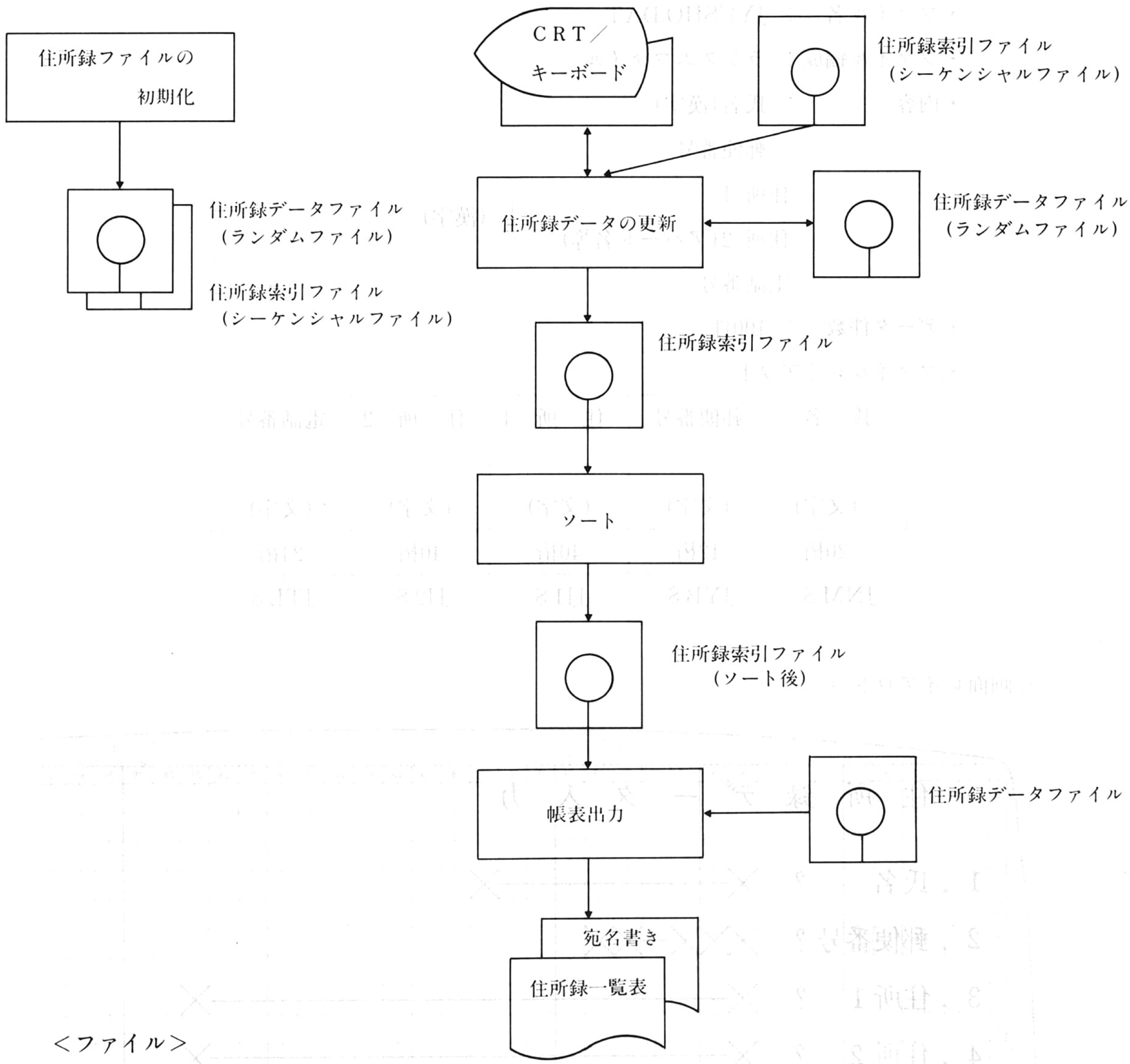
プリンタに住所録の一覧表を印字します。

・宛名書き印字

プリンタに宛名を印字します。

住所録管理のプロセスフローチャート，使用するファイルの形式，画面レイアウト，帳表フォーマットを次に示します。

<プロセスフローチャート>



<ファイル>

◎住所録索引ファイル

・ファイル名　：　JYUSHO.IDX

・ファイル編成：　シーケンシャルファイル

・内容　　　　：　氏名(カタカナ)

　　　　　　　　　レコード番号

・データ件数　：　100件

・ファイルレイアウト(ファイル形式)

氏名 (文字)	レコード番号 (整数)	
INM $	INO	——→ 変数名

◎住所録データファル

・ファイル名　：　JYUSHO.DAT

・ファイル編成：　ランダムファイル

・内容　　　　：　氏名(漢字)

郵便番号

住所 1 ⎫
住所 2(アパート名等) ⎭ (漢字)

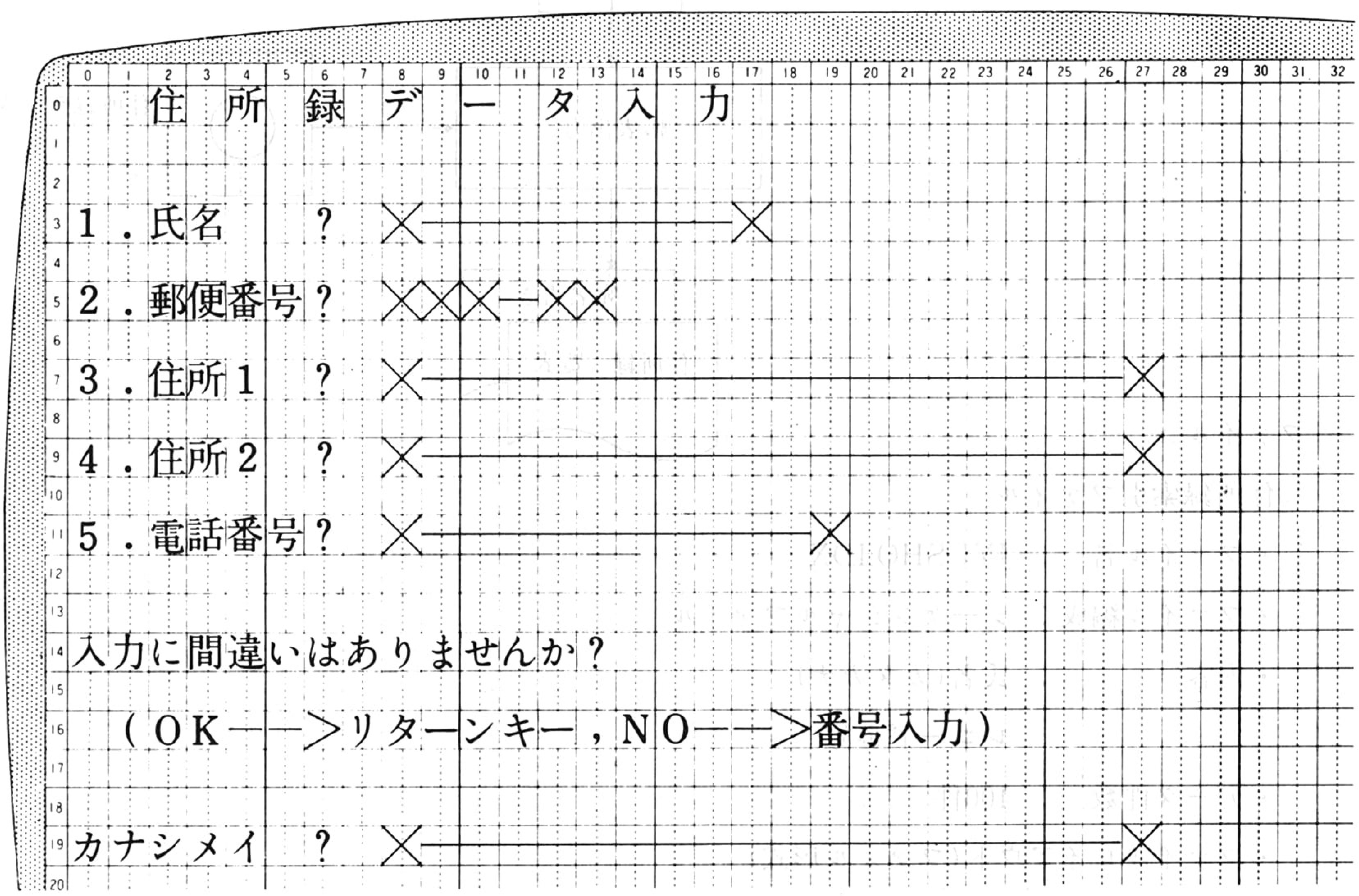
電話番号

・データ件数　：　100件

・ファイルレイアウト

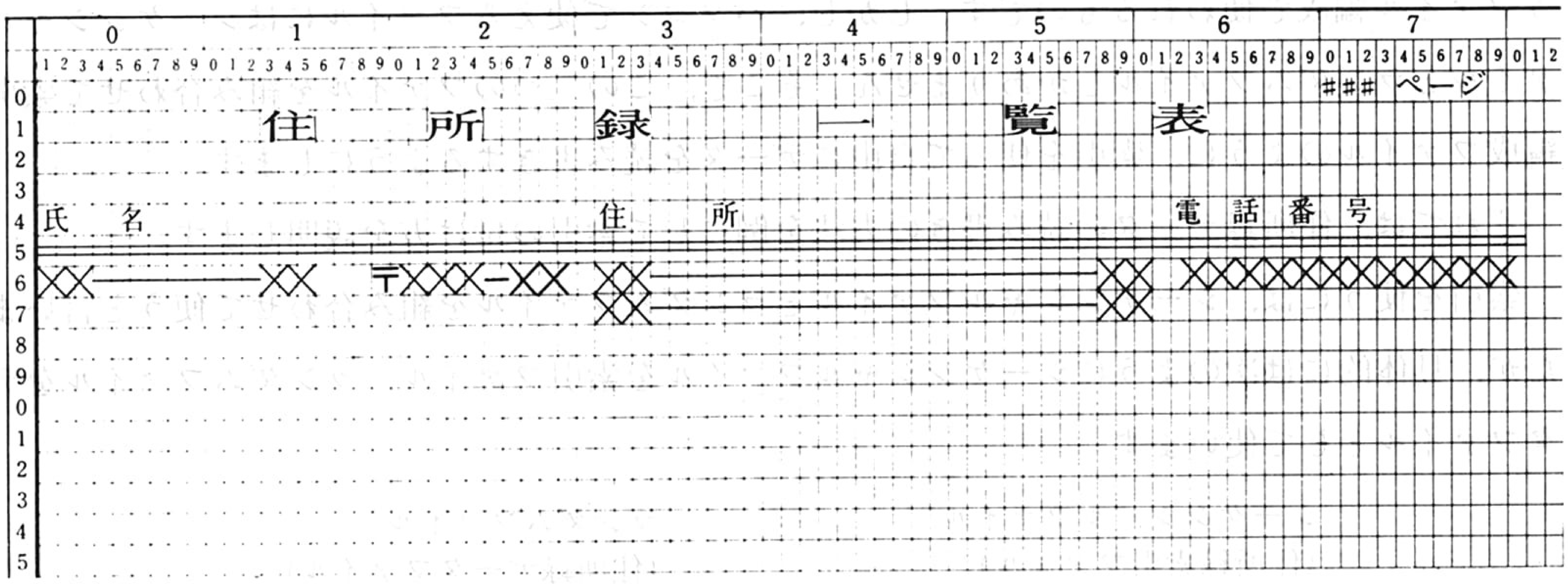
氏　名 (文字)	郵便番号 (文字)	住　所　1 (文字)	住　所　2 (文字)	電話番号 (文字)
20桁	12桁	40桁	40桁	24桁
JNM $	JYB $	JJ1 $	JJ2 $	JTL $

＜画面レイアウト＞



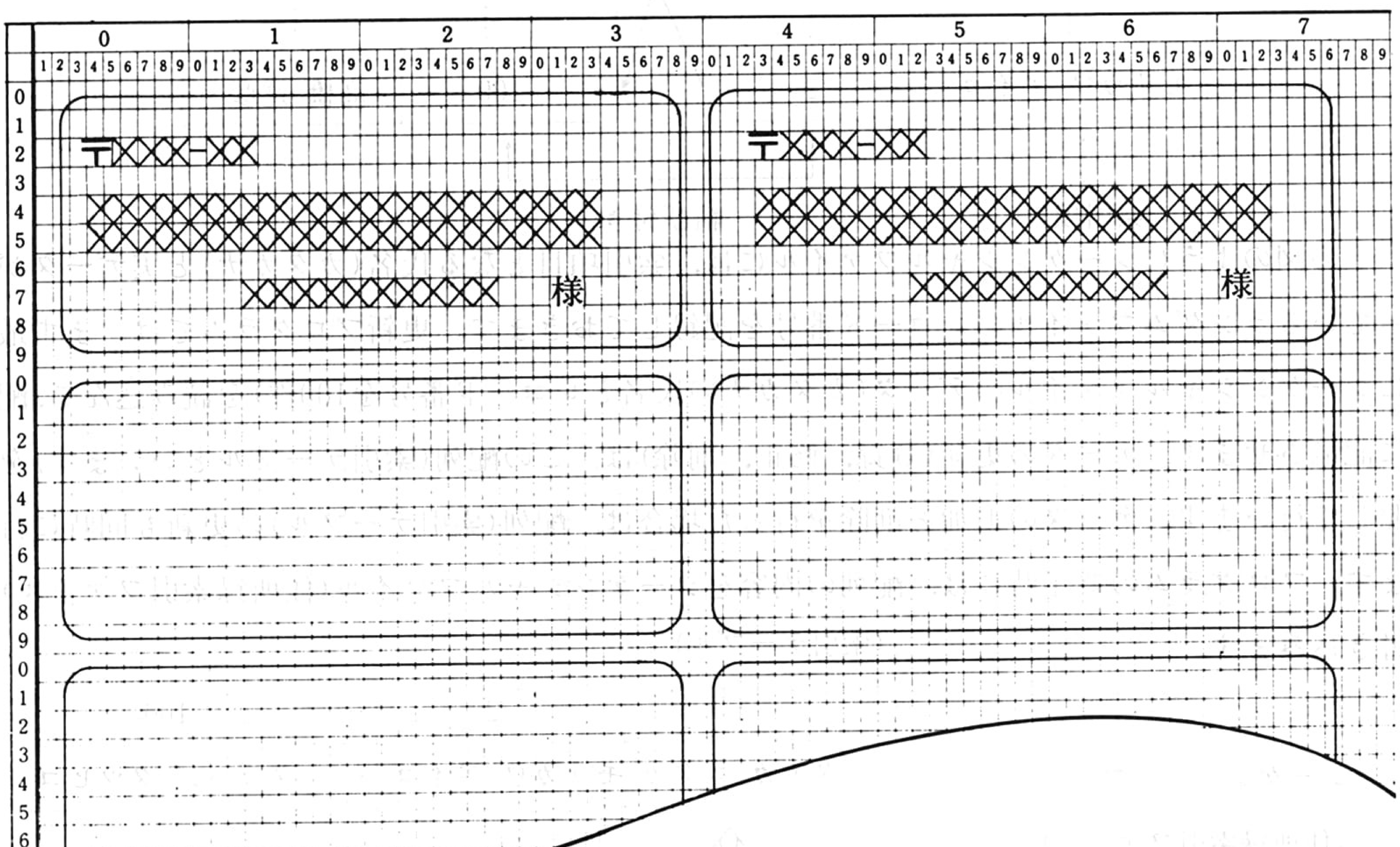
<帳表フォーマット>

・住所録一覧表



・宛名書き(印字用紙は市販のDMラベルを使います)

※ DM : ダイレクトメールの略



1.6.2 索引の付け方

索引を使ってデータを読み書きする方法は，オフコン以上のファイル処理では索引順編成というファイル編成で使われるものです。しかし，パソコンで使えるファイルにはシーケンシャルファイルとランダムファイルしかありません。そこで，この二つのファイルを組み合わせて索引順編成ファイルのように，索引を使って自由にデータを読み書きするようにします。

それでは，住所録データの読み書きの方法を例にして索引の付け方を説明します。

索引を使うには，シーケンシャルファイルとランダムファイルを組み合わせて使うと言いましたが，具体的には次のようにシーケンシャルファイルを索引ファイル，ランダムファイルをデータファイルとして使います。

シーケンシャルファイル
(住所録索引ファイル)

氏名(カタカナ)	レコード番号
タカハシ タツヒコ	100
タナカ ユタカ	1
⋮	⋮
モリカワ アイコ	2

ランダムファイル
(住所録データファイル)

レコード番号	データ
1	田中 豊…
2	森川 愛子…
⋮	⋮
100	高橋 達彦…

索引項目　　結び付き

上の図のようにシーケンシャルファイルには，索引項目となる氏名(カタカナ)と実データが入っているランダムファイルのレコード番号を記録しておきます。更新プログラムでは，まず最初にシーケンシャルファイルのデータ(カタカナの氏名とレコード番号を100件)を読み込んで，配列に記憶させます。データの更新(登録，修正，削除)は，この配列(索引テーブルといいます)を参照して行います。データの追加と削除を行った場合は，配列(索引テーブル)の更新も同時に行います。プログラムの終了時には，配列の内容をシーケンシャルファイル(住所録索引ファイル)に書き込みます。

(索引テーブル)

1	2	….	100
タナカ ユタカ	モリカワ アイコ	….	タカハシ タツヒコ

シーケンシャルファイル
(住所録索引ファイル)

氏名(カタカナ)	レコード番号
タカハシ タツヒコ	100
タナカ ユタカ	1
⋮	⋮
モリカワ アイコ	2

レコード番号を添字にする

レコード番号で読み書きする

ランダムファイル
(住所録データファイル)

レコード番号	データ
1	田中 豊
2	森川 愛子
⋮	⋮
100	高橋 達彦

1.6.3　住所録ファイルの初期化

問題 5　100人分の住所録データが記録できるようにデータファイル(ランダムファイルの初期化と索引ファイル(シーケンシャルファイル)の確保を行って下さい。

【 説 明 】

初期化するデータファイルは,「1.6.1.の＜ファイル＞」で示したような形式を持っています。

ところで，このファイルはランダムファイルであるにもかかわらず識別コードを持っていません。これは，索引ファイルのデータによってレコードの識別を行うことができるからです。また，1件分の長さが136桁であり1セクタ(256桁)の中に複数件のデータを書くことができないので，レコードを効率よく使うといったことはしません。

【 プログラムの作成 】

設問 5-1　次のプログラムをキーインして下さい。

＜キーインするプログラム＞

```
100 ' ジュウショロクファイル イニシャライズ"     (MONDAI.5)
110 *HAJIME:' <<< ショキ ショリ >>>
120  WIDTH 40,20:CLS:PRINT "イニシャライズ" "
130  OPEN "2:JYUSHO.DAT" AS #1
140  FIELD#1,20 AS JNM$,12 AS JYB$,40 AS JJ1$
150  FIELD#1,72 AS DM$,40 AS JJ2$,24 AS JTL$
160  OPEN "2:JYUSHO.IDX" FOR OUTPUT AS #2
170  SP20$=STRING$(40,&H21)
180 *MAIN:' <<< メイン ルーチン >>>
190  GOSUB *DATA.PUT
200 *OWARI:' <<< シュウリョウ ショリ >>>
210  CLOSE
220  WIDTH 80,25:PRINT "イニシャライズ" シュウリョウ"
230  END
240 *DATA.PUT:' <<< デ"ータノセット ト カキコミ >>>
250  LSET JNM$=SP20$
260  LSET JYB$=SP20$
270  LSET JJ1$=SP20$
280  LSET JJ2$=SP20$
290  LSET JTL$=SP20$
300  FOR I=1 TO 100
310    PUT#1,I
320    LOCATE 10,9:PRINT USING "### ハ"ンメ イニシャライス"チュウ";I
330  NEXT I
340  RETURN
```

【 プログラムの実行 】

設問 5-2 正しくキーインできていたら，プログラムを実行して下さい。

＜実行例＞

```
ｲﾆｼｬﾗｲｽﾞ



            36 ﾊﾞﾝﾒ ｲﾆｼｬﾗｲｽﾞﾁｭｳ
```

〔実行後の説明〕

① 実行の経過が目に見えないプログラムは，実行中に不安になることがあります。ここでは，何番目のレコードを初期化しているかを表示して，実行の経過をわかるようにしています。

② 索引ファイルは，ファイルの確保だけで中にはなにも書き込みません。これによってデータファイルには何も入っていないことになります。(索引ファイルのデータは，ランダムファイルの識別コードの代わりになります)

設問 5-3 このプログラムをドライブ 2 のフロッピーディスクへ「MONDAI.5」という名前で保存して下さい。

＜実行結果例＞

```
save "2:MONDAI.5"
Ok
```

1.6.4　索引を使ったデータの更新方法

索引を使ってデータの登録，修正，削除などの更新を行う場合は，次のような手順で処理を行います。

〔手順〕

① 索引テーブルの準備

索引ファイルを読み込んで，配列(索引テーブル)に記憶する。

② 索引項目の入力

データとして索引項目(索引キーといいます)を入力し，その値で索引テーブルを参照します。

③ データの更新

②で索引テーブルを参照した結果，データが存在しない場合は追加処理を行い，データが既に存在している場合は修正または削除処理を行います。

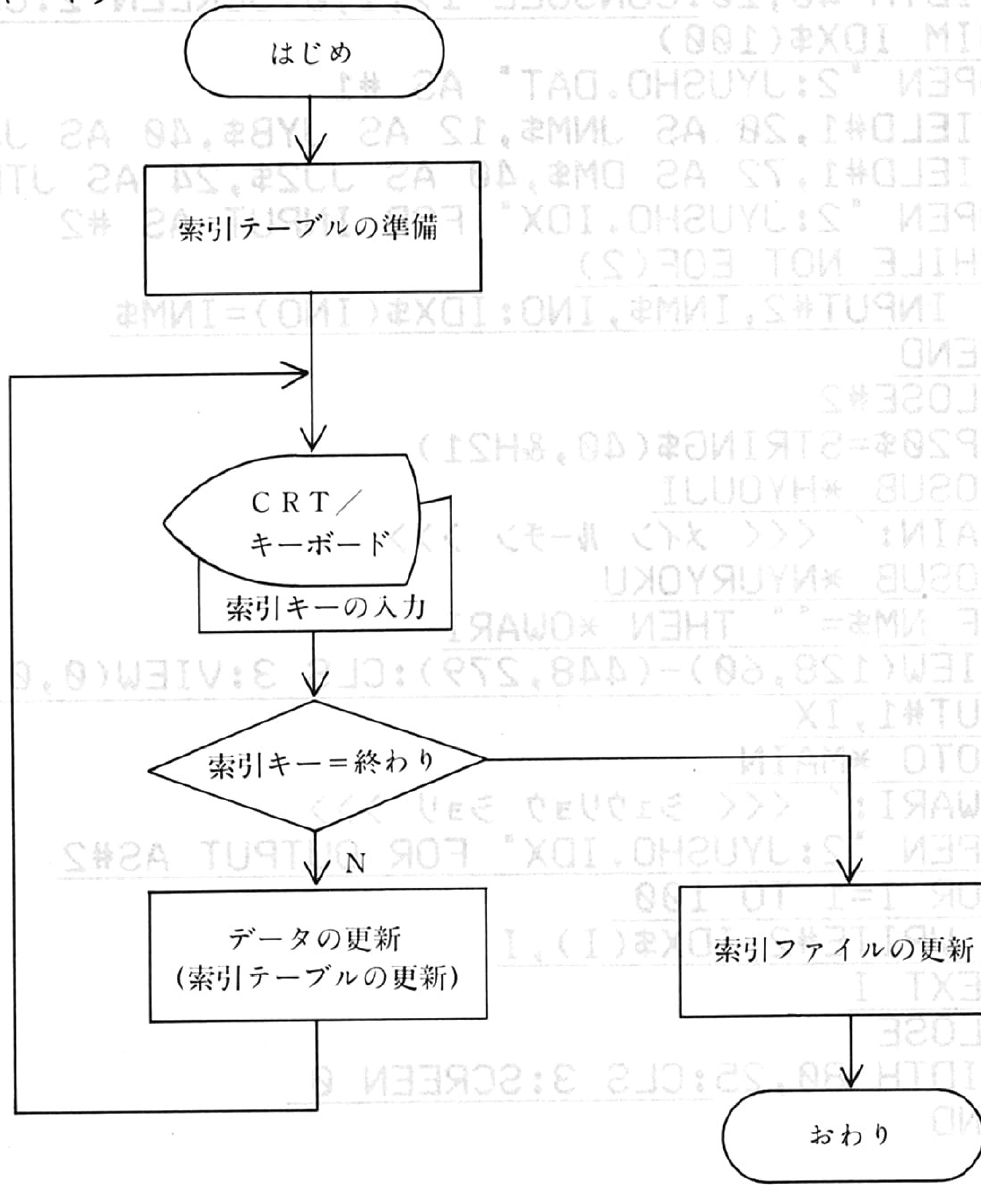
追加処理や削除処理を行った場合は，索引テーブルも更新します。

④ 必要に応じて②，③の処理を繰り返します。

⑤ 索引ファイルの更新

最新の状態になった索引テーブルも更新します。

〔手順のフローチャート〕



問題 6 住所録データの更新を行うプログラムを作成して下さい。

【 説 明 】

前に述べた手順とファイルレイアウトおよび画面レイアウトをもとにしてプログラムを作成します。

【 プログラムの作成 】

設 問 6-1 メモリーに入っているプログラムの行番号240～340を削除して，次の下線部のような修正と追加を行って下さい。

＜修正リスト＞

```
100 ' ｼﾞｭｳｼｮﾛｸﾃﾞｰﾀ ｺｳｼﾝ          (MONDAI.6)
110 *HAJIME:' <<< ｼｮｷ ｼｮﾘ >>>
120  WIDTH 40,20:CONSOLE 19,1,0:SCREEN 2:CLS 3
125  DIM IDX$(100)
130  OPEN "2:JYUSHO.DAT" AS #1
140  FIELD#1,20 AS JNM$,12 AS JYB$,40 AS JJ1$
150  FIELD#1,72 AS DM$,40 AS JJ2$,24 AS JTL$
160  OPEN "2:JYUSHO.IDX" FOR INPUT AS #2
162  WHILE NOT EOF(2)
164    INPUT#2,INM$,INO:IDX$(INO)=INM$
166  WEND
168  CLOSE#2
170  SP20$=STRING$(40,&H21)
175  GOSUB *HYOUJI
180 *MAIN:' <<< ﾒｲﾝ ﾙｰﾁﾝ >>>
182  GOSUB *NYURYOKU
184  IF NM$="" THEN *OWARI
186  VIEW(128,60)-(448,279):CLS 3:VIEW(0,0)-(639,399)
188  PUT#1,IX
190  GOTO *MAIN
200 *OWARI:' <<< ｼｭｳﾘｮｳ ｼｮﾘ >>>
202  OPEN "2:JYUSHO.IDX" FOR OUTPUT AS#2
204  FOR I=1 TO 100
206    WRITE#2,IDX$(I),I
208  NEXT I
210  CLOSE
220  WIDTH 80,25:CLS 3:SCREEN 0
230  END
```

〔修正後の説明〕

① 修正を行った部分は，初期処理，処理の流れ(メインルーチン)，終了処理を行う部分です。

② 行番号162～166は，索引ファイルのデータを読み込んで索引テーブルを作成しています。

③ 行番号175で実行しているサブルーチン(＊HYOUJI)は，画面レイアウト(「1.6.1」参照)に従ってそのメッセージを表示するものです。

④ 行番号182で実行しているサブルーチン(＊NYURYOKU)は，データの更新と索引テーブルの更新を行うものです。

⑤ 行番号184は、索引キー(カナ氏名)の入力がなかったとき終了処理へ分岐します。

⑥ 行番号188は，VIEWという命令(「2.5.2 スクリーン座標系とは」の【 命令のまとめ 】を参照)を使って漢字データの入力エリアをクリアします。

⑦ 行番号188は，住所録データの書き込みを行います。ここでは，便宜的に削除するデータもランダムファイルへの書き込みを行います。(削除データの識別は索引テーブルで行います)

⑧ 行番号202～208は，索引ファイルの更新を行います。プログラム中の索引テーブルの添字とデータのレコード番号は一致しています。したがって，索引ファイルを更新するときのレコード番号は添字(I)で構いません。

※ 索引テーブルを読むとき，レコード番号と添字の値は一致していない場合があります。
　これは，帳表出力を行うときに索引ファイルのソートを行うためです。

設問 6-2 次のサブルーチンと漢字コードデータをキーインして下さい。

＜サブルーチン＞

```
1000 '
1010 *HYOUJI:' <<< ｶﾞﾒﾝ ﾋｮｳｼﾞ >>>
1020   X=2:Y=0 :XE=16:GOSUB *KANJI.PUT
1030   X=0:Y=3 :XE=6 :GOSUB *KANJI.PUT
1040   X=0:Y=5 :XE=6 :GOSUB *KANJI.PUT
1050   X=0:Y=7 :XE=6 :GOSUB *KANJI.PUT
1060   X=0:Y=9 :XE=6 :GOSUB *KANJI.PUT
1070   X=0:Y=11:XE=6 :GOSUB *KANJI.PUT
1080   X=0:Y=14:XE=13:GOSUB *KANJI.PUT
1090   X=1:Y=16:XE=23:GOSUB *KANJI.PUT
1100   RETURN
2000  *KANJI.PUT:' << ｶﾝｼﾞ ｼｭﾂﾘｮｸ >>
2010    Y=Y*20
2020    FOR X=X TO XE
2030      READ KN$:KN%=VAL("&h"+KN$)
2040      PUT (X*16,Y),KANJI(KN%)
2050    NEXT X
2060    RETURN
```

＜漢字コードデータ＞

```
10000 DATA 3D3B,2121,3D6A,2121,4F3F,2121,2547,2121
10010 DATA 213D,2121,253F,2121,467E,2121,4E4F
10020 DATA 2331,2125,3B61,4C3E,2121,2121,2129
10030 DATA 2332,2125,4D39,4A58,4856,3966,2129
10040 DATA 2333,2125,3D3B,3D6A,2331,2121,2129
10050 DATA 2334,2125,3D3B,3D6A,2332,2121,2129
10060 DATA 2335,2125,4545,4F43,4856,3966,2129
10070 DATA 467E,4E4F,244B,3456,3063,2424,244F
10080 DATA 2422,246A,245E,243B,2473,242B,2129
10090 DATA 214A,234F,234B,215D,215D,2153,256A,253F
10100 DATA 213D,2573,252D,213D,2124,234E,234F,215D
10110 DATA 215D,2153,4856,3966,467E,4E4F,214B
```

〔キーイン後の説明〕

① キーインしたサブルーチン(＊HYOUJI)は，画面レイアウトに従ってメッセージを表示させるものです。

「＊KANJI.PUT」は，漢字コードデータの読み込みと漢字の表示を「＊HYOUJI」からの指示(変数)に従って行います。

変数 X …… メッセージの先頭の列位置
(漢字は表示位置をドット単位で指定し，表示したときの大きさが縦16ドット・横16ドットなので，実際の位置は「X＊16」になります)

変数 Y …… メッセージの行位置
(漢字表示の行間として4ドットあけます。したがって実際の位置は「Y＊(16＋4)」になります)

変数XE…… メッセージ終了の列位置

設問 6-3 次のサブルーチンをキーインして下さい。

＜キーインするサブルーチン＞

```
3000 '
3010 *NYURYOKU:' <<< ﾃﾞｰﾀ ﾆｭｳﾘｮｸ >>>
3020   CLS:LOCATE 0,19:INPUT "ｶﾅｼﾒｲ ";NM$
3030   IF NM$="" THEN GOTO 3180
3040   GOSUB *DATA.GET
3050   IF IDX$(IX)="" THEN IDX$(IX)=NM$ ELSE GOTO 3110
3060   GOSUB *N1
3070   GOSUB *N2
3080   GOSUB *N3
```

(つづく)

```
3090   GOSUB *N4
3100   GOSUB *N5
3110   LOCATE 15,14:PRINT "■"
3120   LOCATE 15,14:YN$=INKEY$:IF YN$="" THEN 3120
3130   LOCATE 15,14:PRINT YN$
3140   IF YN$="0" THEN IDX$(IX)="":GOTO 3180
3150   IF YN$=CHR$(&HD) THEN 3180
3160   ON VAL(YN$) GOSUB *N1,*N2,*N3,*N4,*N5
3170   GOTO 3110
3180   LOCATE 15,14:PRINT " "
3190   RETURN
4000  *DATA.GET:' < ﾃﾞｰﾀ ﾖﾐｺﾐ >
4010    FOR IX=1 TO 100
4020      IF IDX$(IX)=NM$ THEN 4060
4030      IF IDX$(IX)="" THEN IX2=IX
4040    NEXT IX
4050    IX=IX2:GOTO 4120
4060    GET#1,IX
4070    Y=3 :KN$=JNM$:GOSUB *DATA.HYOUJI
4080    Y=5 :KN$=JYB$:GOSUB *DATA.HYOUJI
4090    Y=7 :KN$=JJ1$:GOSUB *DATA.HYOUJI
4100    Y=9 :KN$=JJ2$:GOSUB *DATA.HYOUJI
4110    Y=11:KN$=JTL$:GOSUB *DATA.HYOUJI
4120    RETURN
5000   *DATA.HYOUJI:' ﾃﾞｰﾀ ﾋｮｳｼﾞ
5010     X=8
5020     FOR I=1 TO LEN(KN$) STEP 2
5030       PUT (X*16,Y*20),KANJI(CVI(MID$(KN$,I,2)))
5040       X=X+1
5050     NEXT I
5060     RETURN
6000  *N1:Y=3 :XE=17:GOSUB *KANJI.GET
6010    KC$=KC$+SP20$:LSET JNM$=KC$:RETURN
6020  *N2:Y=5 :XE=13:GOSUB *KANJI.GET
6030    KC$=KC$+SP20$:LSET JYB$=KC$:RETURN
6040  *N3:Y=7 :XE=27:GOSUB *KANJI.GET
6050    KC$=KC$+SP20$:LSET JJ1$=KC$:RETURN
6060  *N4:Y=9 :XE=27:GOSUB *KANJI.GET
6070    KC$=SP20$+KC$:LSET JJ2$=RIGHT$(KC$,40):RETURN
6080  *N5:Y=11:XE=19:GOSUB *KANJI.GET
6090    KC$=KC$+SP20$:LSET JTL$=KC$:RETURN
```

〔キーイン後の説明〕

① 「＊NYURYOKU」は，データ更新処理の流れを制御する部分です。

・行番号3020～3050は，索引キー(カナ氏名)を入力して，該当するデータがない場合はデータ入力を行うようにし，該当するデータがある場合は修正番号を入力するようにします。

・行番号3060～3100は，住所録データの入力を行います。

・行番号3110～3170は，修正番号を入力してその項目のデータを入力します。修正番号として「0」を入力した場合は，削除を行います。

② 「＊DATA.GET」は、索引テーブルの検索とデータの読み込みを行います。

・行番号4010～4050は，入力したカナ氏名が索引テーブルにあるか検索します。

・行番号4060～4110は，索引テーブルにカナ氏名があった場合にデータを読み込んでその内容を表示します。

③ 「＊DATA.HYOUJI」は，読み込んだ漢字データを表示します。

④ 「＊N1～＊N5」の各サブルーチンは，各項目の入力を行います。入力したデータの余白には空白を入れてランダムバッファにセットします。このうち，「住所2」については右寄せで表示するように空白を頭に付けます。

設問 6-4 次のサブルーチンをキーインして下さい。

＜キーインするサブルーチン＞

```
7000    *KANJI.GET:' < ｶﾝｼﾞ ﾆｭｳﾘｮｸ >
7010      KC$=""
7020      FOR X=8 TO XE
7030        LOCATE X-1,Y:PRINT " ";
7040        LOCATE X,Y:PRINT "■";
7050        KN$=""
7060        A$=INKEY$:IF A$="" THEN 7060
7070        IF A$=CHR$(&H7F) THEN GOSUB *K.DEL:GOTO 7040
7080        IF A$=CHR$(&HD)  THEN X=X+1:GOTO 7180
7090        IF A$=CHR$(&H20) THEN KN$="2121":GOTO 7130
7100        IF A$=CHR$(&HC)  THEN GOSUB *K.CLR:GOTO *KANJI.GET
7110        KN$=KN$+A$
7120        IF LEN(KN$)><4 THEN 7060
7130        KN%=VAL("&h"+KN$)
7140        IF KN%=0 THEN BEEP:GOTO 7040
7150        PUT (X*16,Y*20),KANJI(KN%)
7160        KC$=KC$+MKI$(KN%)
7170      NEXT X
7180      LOCATE X-1,Y:PRINT " "
7190      RETURN
8000     *K.DEL:' ｶﾝｼﾞ ﾄﾘｹｼ
8010       IF LEN(KN$)>0 THEN 8070
8020       IF X=8 THEN KC$="":GOTO 8070
8030       X=X-1
8040       KC$=LEFT$(KC$,LEN(KC$)-2)
8050       LOCATE X+1,Y:PRINT " ";
8060       PUT (X*16,Y*20),KANJI(&H2121),PSET
8070       RETURN
9000     *K.CLR:' ｶﾝｼﾞ ｸﾘｱ
9010       KC$=LEFT$(SP20$,(XE-7)*2):KN$=""
9020       FOR I=XE+1 TO 8 STEP -1
9030         X=I:GOSUB *K.DEL
9040       NEXT I
9050       RETURN
```

〔キーイン後の説明〕

① キーインしたサブルーチンは，漢字データの入力を行うものです。

② これは，「1.3.7」のMONDAI.K4の漢字氏名の入力とほぼ同じですが，新しく［HOME CLR］キーに対する処理(その項目をクリア)が追加され(行番号7100，9000～9050)，漢字データを記憶する変数をKC＄(MONDAI.K4ではNM＄)に変更しています。

【 プログラムの確認 】

設問 6-5 プログラムのリストをプリンタに印字して正しくキーインできているかどうか確認して下さい。

<確認用リスト>

```
100 ′ ｼﾞｭｳｼｮﾛｸﾃﾞｰﾀ ｺｳｼﾝ                (MONDAI.6)
110 *HAJIME:′ <<< ｼｮｷ ｼｮﾘ >>>
120  WIDTH 40,20:CONSOLE 19,1,0:SCREEN 2:CLS 3
125  DIM IDX$(100)
130  OPEN "2:JYUSHO.DAT" AS #1
140  FIELD#1,20 AS JNM$,12 AS JYB$,40 AS JJ1$
150  FIELD#1,72 AS DM$,40 AS JJ2$,24 AS JTL$
160  OPEN "2:JYUSHO.IDX" FOR INPUT AS #2
162  WHILE NOT EOF(2)
164    INPUT#2,INM$,INO:IDX$(INO)=INM$
166  WEND
168  CLOSE#2
170  SP20$=STRING$(40,&H21)
175  GOSUB *HYOUJI
180 *MAIN:′ <<< ﾒｲﾝ ﾙｰﾁﾝ >>>
182  GOSUB *NYURYOKU
184  IF NM$="" THEN *OWARI
186  VIEW(128,60)-(448,279):CLS 3:VIEW(0,0)-(639,399)
188  PUT#1,IX
190  GOTO *MAIN
200 *OWARI:′ <<< ｼｭｳﾘｮｳ ｼｮﾘ >>>
202  OPEN "2:JYUSHO.IDX" FOR OUTPUT AS#2
204  FOR I=1 TO 100
206    WRITE#2,IDX$(I),I
208  NEXT I
210  CLOSE
220  WIDTH 80,25:CLS 3:SCREEN 0
230  END
1000 ′
1010 *HYOUJI:′ <<< ｶﾞﾒﾝ ﾋｮｳｼﾞ >>>
1020   X=2:Y=0 :XE=16:GOSUB *KANJI.PUT
1030   X=0:Y=3 :XE=6 :GOSUB *KANJI.PUT
1040   X=0:Y=5 :XE=6 :GOSUB *KANJI.PUT
1050   X=0:Y=7 :XE=6 :GOSUB *KANJI.PUT
1060   X=0:Y=9 :XE=6 :GOSUB *KANJI.PUT
1070   X=0:Y=11:XE=6 :GOSUB *KANJI.PUT
1080   X=0:Y=14:XE=13:GOSUB *KANJI.PUT
1090   X=1:Y=16:XE=23:GOSUB *KANJI.PUT
1100   RETURN
```

(つづく)

```
2000  *KANJI.PUT:' << ｶﾝｼﾞ ｼｭﾂﾘｮｸ >>
2010    Y=Y*20
2020    FOR X=X TO XE
2030      READ KN$:KN%=VAL("&h"+KN$)
2040      PUT (X*16,Y),KANJI(KN%)
2050    NEXT X
2060    RETURN
3000 '
3010 *NYURYOKU:' <<< ﾃﾞｰﾀ ﾆｭｳﾘｮｸ >>>
3020   CLS:LOCATE 0,19:INPUT "ｶﾅｼﾒｲ ";NM$
3030   IF NM$="" THEN GOTO 3180
3040   GOSUB *DATA.GET
3050   IF IDX$(IX)="" THEN IDX$(IX)=NM$ ELSE GOTO 3110
3060   GOSUB *N1
3070   GOSUB *N2
3080   GOSUB *N3
3090   GOSUB *N4
3100   GOSUB *N5
3110   LOCATE 15,14:PRINT "■"
3120   LOCATE 15,14:YN$=INKEY$:IF YN$="" THEN 3120
3130   LOCATE 15,14:PRINT YN$
3140   IF YN$="0" THEN IDX$(IX)="":GOTO 3180
3150   IF YN$=CHR$(&HD) THEN 3180
3160   ON VAL(YN$) GOSUB *N1,*N2,*N3,*N4,*N5
3170   GOTO 3110
3180   LOCATE 15,14:PRINT " "
3190   RETURN
4000  *DATA.GET:' < ﾃﾞｰﾀ ﾖﾐｺﾐ >
4010    FOR IX=1 TO 100
4020      IF IDX$(IX)=NM$ THEN 4060
4030      IF IDX$(IX)="" THEN IX2=IX
4040    NEXT IX
4050    IX=IX2:GOTO 4120
4060    GET#1,IX
4070    Y=3 :KN$=JNM$:GOSUB *DATA.HYOUJI
4080    Y=5 :KN$=JYB$:GOSUB *DATA.HYOUJI
4090    Y=7 :KN$=JJ1$:GOSUB *DATA.HYOUJI
4100    Y=9 :KN$=JJ2$:GOSUB *DATA.HYOUJI
4110    Y=11:KN$=JTL$:GOSUB *DATA.HYOUJI
4120    RETURN
5000   *DATA.HYOUJI:' ﾃﾞｰﾀ ﾋｮｳｼﾞ
5010     X=8
5020     FOR I=1 TO LEN(KN$) STEP 2
5030       PUT (X*16,Y*20),KANJI(CVI(MID$(KN$,I,2)))
5040       X=X+1
5050     NEXT I
5060     RETURN
6000  *N1:Y=3 :XE=17:GOSUB *KANJI.GET
6010    KC$=KC$+SP20$:LSET JNM$=KC$:RETURN
6020  *N2:Y=5 :XE=13:GOSUB *KANJI.GET
6030    KC$=KC$+SP20$:LSET JYB$=KC$:RETURN
6040  *N3:Y=7 :XE=27:GOSUB *KANJI.GET
6050    KC$=KC$+SP20$:LSET JJ1$=KC$:RETURN
6060  *N4:Y=9 :XE=27:GOSUB *KANJI.GET
6070    KC$=SP20$+KC$:LSET JJ2$=RIGHT$(KC$,40):RETURN
6080  *N5:Y=11:XE=19:GOSUB *KANJI.GET
6090    KC$=KC$+SP20$:LSET JTL$=KC$:RETURN
```

（つづく）

```
7000    *KANJI.GET:' < ｶﾝｼﾞ ﾆｭｳﾘｮｸ >
7010      KC$=""
7020      FOR X=8 TO XE
7030        LOCATE X-1,Y:PRINT " ";
7040        LOCATE X,Y:PRINT "■";
7050        KN$=""
7060        A$=INKEY$:IF A$="" THEN 7060
7070        IF A$=CHR$(&H7F) THEN GOSUB *K.DEL:GOTO 7040
7080        IF A$=CHR$(&HD)  THEN X=X+1:GOTO 7180
7090        IF A$=CHR$(&H20) THEN KN$="2121":GOTO 7130
7100        IF A$=CHR$(&HC)  THEN GOSUB *K.CLR:GOTO *KANJI.GET
7110        KN$=KN$+A$
7120        IF LEN(KN$)><4 THEN 7060
7130        KN%=VAL("&h"+KN$)
7140        IF KN%=0 THEN BEEP:GOTO 7040
7150        PUT (X*16,Y*20),KANJI(KN%)
7160        KC$=KC$+MKI$(KN%)
7170      NEXT X
7180      LOCATE X-1,Y:PRINT " "
7190      RETURN
8000     *K.DEL:' ｶﾝｼﾞ ﾄﾘｹｼ
8010       IF LEN(KN$)>0 THEN 8070
8020       IF X=8 THEN KC$="":GOTO 8070
8030       X=X-1
8040       KC$=LEFT$(KC$,LEN(KC$)-2)
8050       LOCATE X+1,Y:PRINT " ";
8060       PUT (X*16,Y*20),KANJI(&H2121),PSET
8070       RETURN
9000     *K.CLR:' ｶﾝｼﾞ ｸﾘｱ
9010       KC$=LEFT$(SP20$,(XE-7)*2):KN$=""
9020       FOR I=XE+1 TO 8 STEP -1
9030         X=I:GOSUB *K.DEL
9040       NEXT I
9050       RETURN
10000 DATA 3D3B,2121,3D6A,2121,4F3F,2121,2547,2121
10010 DATA 213D,2121,253F,2121,467E,2121,4E4F
10020 DATA 2331,2125,3B61,4C3E,2121,2121,2129
10030 DATA 2332,2125,4D39,4A58,4856,3966,2129
10040 DATA 2333,2125,3D3B,3D6A,2331,2121,2129
10050 DATA 2334,2125,3D3B,3D6A,2332,2121,2129
10060 DATA 2335,2125,4545,4F43,4856,3966,2129
10070 DATA 467E,4E4F,244B,3456,3063,2424,244F
10080 DATA 2422,246A,245E,243B,2473,242B,2129
10090 DATA 214A,234F,234B,215D,215D,2153,256A,253F
10100 DATA 213D,2573,252D,213D,2124,234E,234F,215D
10110 DATA 215D,2153,4856,3966,467E,4E4F,214B
```

【 プログラムの実行 】

設問 6-6 正しくキーインできていたら，プログラムを実行して次のデータを登録して下さい。

＜登録データ＞

氏　名	郵便番号	住　　所　1	住　　所　2	電話番号
高橋 達彦	243-02	厚木市飯山1000		0462-20-0123
原田 一朗	251	藤沢市藤沢7		0466-23-1111
山田 多郎	363	桶川市鴨川10-4		0487-63-8000
渡辺 次郎	211	川崎市中原区新城3-26	第一男子寮	044-384-5678
河合 由絵	215	川崎市麻生区高石88	高石荘2-2	
松下 紀子	186	国立市東3-9		0425-77-7777
森川 愛子	182	調布市柴崎1-11	柴崎マンション802	0424-88-1234
田中 豊	151	渋谷区本町3-48		03-378-9876
佐藤 純子	214	川崎市多摩区生田8-11		044-931-3838
鈴木 宏子	168	杉並区宮前4-5-24		03-333-3333

＜漢字コード＞

漢字	コード	漢字	コード	漢字	コード	漢字	コード	漢字	コード	漢字	コード	漢字	コード	漢字	コード
1	2331	厚	387C	田	4544	中	4366	市	3B54	森	3F39	中	4366	摩	4B60
2	2332	木	4C5A	多	423F	原	3836	麻	4B63	川	406E	豊	4B2D	区	3668
3	2333	市	3B54	郎	4F3A	区	3668	生	4038	愛	3026	渋	3D42	生	4038
4	2334	飯	4853	桶	3233	新	3F37	区	3668	子	3B52	谷	432B	田	4544
5	2335	山	3B33	川	406E	城	3E6B	高	3762	調	4434	区	3668	鈴	4E6B
6	2336	原	3836	市	3B54	第	4268	石	4050	布	495B	本	4B5C	木	4C5A
7	2337	田	4544	鴨	337B	男	434B	荘	4171	市	3B54	町	442E	宏	3928
8	2338	一	306C	川	406E	子	3B52	松	3E3E	柴	3C46	佐	3A34	子	3B52
9	2339	朗	4F2F	渡	454F	寮	4E40	下	323C	崎	3A6A	藤	4623	杉	3F79
0	2330	藤	4623	辺	4A55	河	324F	紀	352A	マ	255E	純	3D63	並	4A42
—	215D	沢	4274	次	3C21	合	3967	子	3B52	ン	2573	子	3B52	区	3668
高	3962	市	3B54	郎	4F3A	由	4D33	国	3971	シ	2537	川	406E	宮	355C
橋	3636	藤	4623	川	406E	絵	3328	立	4E29	ョ	2567	崎	3A6A	前	4130
達	4323	沢	4274	崎	3A6A	川	406E	市	3B54	ン	2573	市	3B54		
彦	4927	山	3B33	市	3B54	崎	3A6A	東	456C	田	4544	多	423F		

※ 漢字コードを正しく入力したにもかかわらず表示がおかしい場合は，カナロックキーが押されていないかどうかを確認して下さい。

＜実行例＞

```
  住 所 録 デ ー タ 入 力

 1. 氏名    ? 高橋 達彦

 2. 郵便番号? 243-02

 3. 住所1  ? 厚木市飯山1000

 4. 住所2  ?

 5. 電話番号? 0462-20-0123

 入力に間違いはありませんか? ■

  (OK--〉リターンキー, NO--〉番号入力)
```

〔実行後の説明〕

① データ入力のとき，初めに必ず索引キーとなるカナ氏名の入力を行います。(画面の最下行で入力する)

設問 6-7 このプログラムをドライブ 2 のフロッピーディスクへ「MONDAI.6」という名前で保存して下さい。

＜実行結果例＞

```
save "MONDAI.6"
Ok
```

1.6.5 索引ファイルのソート

帳表出力を行う際に，データが氏名の五十音順に出力されるように索引ファイルをソートします。

問題 7 索引ファイルをカナ氏名の五十音順にソートして下さい。

【説 明】

漢字データを五十音順に並べるには，その読み方をソートキーとしてソートします。ここで扱う住所録の索引は，べつにカナ氏名でなくても各データを識別できるものであれば何でもよいのですが，氏名を索引にしたいということと氏名の五十音順に帳表を出力するという目的からカナの氏名を索引にしました。

【プログラムの作成】

設問 7-1 メモリーをクリアしてから次のプログラムをキーインして下さい。

＜キーインするプログラム＞

```
100 ' ｻｸｲﾝ ｿｰﾄ
110 *HAJIME:' <<< ｼｮｷ ｼｮﾘ >>>
120  N=100:DIM INM$(N),INO(N)
130  WIDTH 40,20:CLS
140  OPEN "2:JYUSHO.IDX" FOR INPUT AS#1
150  LOCATE 10,10:PRINT "ｻｸｲﾝ ﾖﾐｺﾐ"
160  FOR I=1 TO N
170    IF EOF(1) THEN 200
180    INPUT#1,INM$(I),INO(I)
190  NEXT I
200  CLOSE
210 '
220 *SORT:' <<< ｿｰﾄ >>>
230  FOR I=1 TO N-1
240    FOR J=I+1 TO N
250      IF INM$(I)<=INM$(J) THEN 280
260      SWAP INM$(I),INM$(J)
270      SWAP INO(I),INO(J)
280    NEXT J
290  LOCATE 10,10:PRINT USING "### ｹﾝ  ｿｰﾄｽﾞﾐ";I
300  NEXT I
310 '
320 *OWARI:' <<< ｼｭｳﾘｮｳ ｼｮﾘ >>>
330  OPEN "2:JYUSHO.IDX" FOR OUTPUT AS #1
340  CLS:LOCATE 10,10:PRINT "ｻｸｲﾝ ｶｷｺﾐ"
350  FOR I=1 TO N
360    WRITE#1,INM$(I),INO(I)
370  NEXT I
380  CLOSE
390  WIDTH 80,25
400  END
```

〔キーイン後の説明〕

① キーインしたプログラムは，索引ファイルのソートを逐次決定法(上巻「4章」参照)で行うものです。

② ソート中は，何件のソートが終了したかを表示するようにします。

③ データの最大件数を変数Nで指定するようにしています。したがって，索引ファイルのデータ件数に変動があった場合は，行番号120で変数Nの値を変えるだけですみます。

【 プログラムの実行 】

設問 7-2 正しくキーインできているかどうか確認した後，プログラムを実行して下さい。

<実行例>

```
39 ケン  ソートズ゛ミ
```

〔実行後の説明〕

① 表示されるソート件数は，データ件数よりも1少ない数(99)で終わります。

設問 7-3 このプログラムをドライブ 2 のフロッピーディスクへ「MONDAI.7」という名前で保存して下さい。

<実行結果例>

```
save "2:MONDAI.7"
Ok
```

1.6.6 住所録一覧表の出力

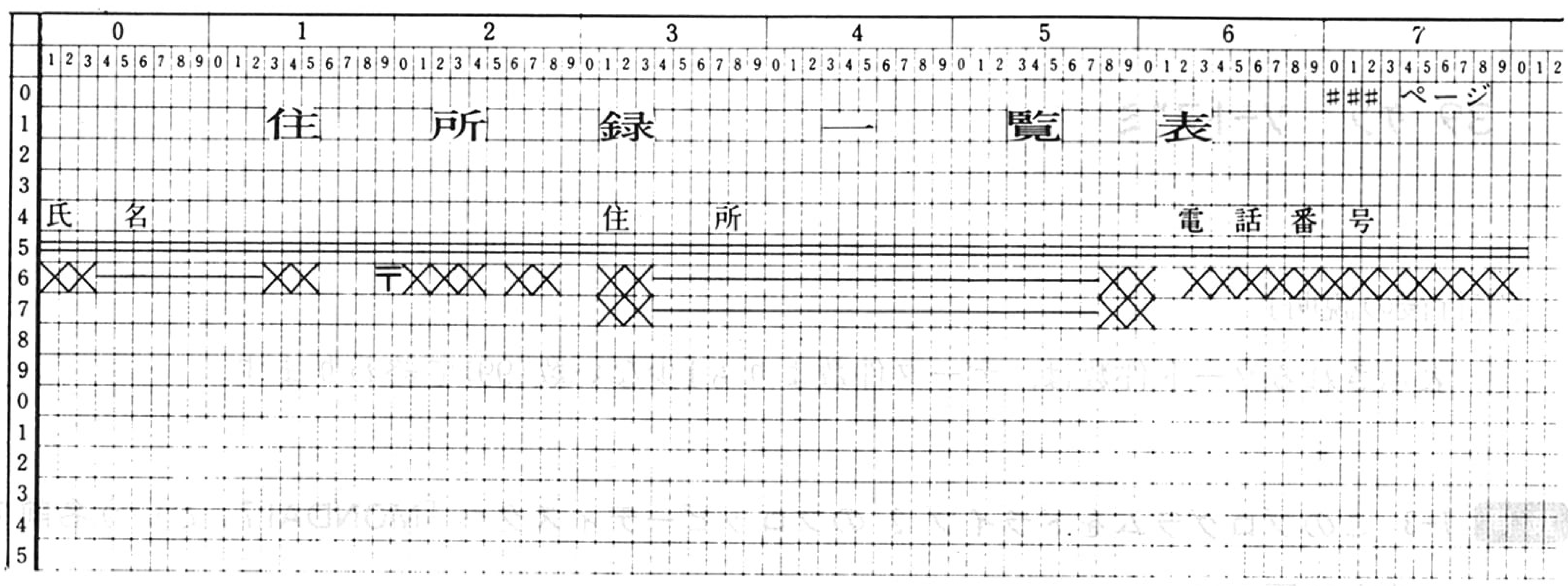
住所録データの一覧表をプリンタに印字します。

> **問題 8** 索引ファイルを使って住所録データを読み込み,「1.6.1」の帳表フォーマットに従って住所録一覧表を出力しなさい。

【説 明】

「1.6.1」で示した住所録一覧表の帳表フォーマットを次に示しておきます。

＜住所録一覧表の帳表フォーマット＞



・1ページに印字する住所録データは20件とします。

・プログラムは,「1.3.7」で作成したプログラム(MONDAI.K3)を参考にして作成します。

【 プログラムの作成 】

設問 8-1 メモリーをクリアしてから，次のプログラムをキーインして下さい。

<キーインするプログラム>

```
100 ' ｼﾞｭｳｼｮﾛｸ ｲﾁﾗﾝﾋｮｳ            (MONDAI.8)
110 *HAJIME:' <<< ｼｮｷ ｼｮﾘ >>>
120  DIM INM$(100),INO(100)
130  OPEN "2:JYUSHO.DAT" AS #1
140  FIELD#1,20 AS JNM$,12 AS JYB$,40 AS JJ1$
150  FIELD#1,72 AS DM$,40 AS JJ2$,24 AS JTL$
160  OPEN "2:JYUSHO.IDX" FOR INPUT AS #2
170  FOR I=1 TO 100
180    IF EOF(2) THEN 200 ELSE INPUT#2,INM$(I),INO(I)
190  NEXT I
200 *MAIN:' <<< ﾒｲﾝ ﾙｰﾁﾝ >>>
210  GOSUB *MIDASHI
220  FOR IX=1 TO 100
230    GOSUB *MEISAI
240  NEXT IX
250 *OWARI:' <<< ｼｭｳﾘｮｳ ｼｮﾘ >>>
260  CLOSE
270  END
```

〔キーイン後の説明〕

① 初期処理で索引テーブルを作成していますが，更新(「1.6.4」参照)のときと違ってレコード番号は添字の値と一致しません。したがって，レコード番号も配列(索引テーブル)に入れるようにします。

② メインルーチンでは，見出しの印字(＊MIDASHI)を行った後，変数IXを使って100件の明細を印字(＊MEISAI)しています。

設問 8-2 次のサブルーチンをキーインして下さい。

<キーインするサブルーチン>

```
1000 '
1010 *MEISAI:' <<< ﾒｲｻｲ ｲﾝｼﾞ ｼｮﾘ >>>
1020  IF INM$(IX)="" THEN 1140
1030  GET#1,INO(IX)
1040  KJ=22:KN$=JNM$:GOSUB *KANJI.INJI
1050  LPRINT CHR$(&H1B);"K";CHR$(&H22);CHR$(&H29);CHR$(&H1B);"H";
1060  KJ=14:KN$=JYB$:GOSUB *KANJI.INJI
1070  KJ=42:KN$=JJ1$:GOSUB *KANJI.INJI
1080  KJ=24:KN$=JTL$:GOSUB *KANJI.INJI
1090  LPRINT:LPRINT SPC(30);
1100  KJ=40:KN$=JJ2$:GOSUB *KANJI.INJI
```

(つづく)

```
1110  LPRINT:LPRINT
1120  LC=LC+1
1130  IF LC=20 THEN GOSUB *MIDASHI:LC=0
1140  RETURN
2000 *KANJI.INJI:´ << ｶﾝｼﾞｲﾝｼﾞ >>
2010  LPRINT CHR$(&H1B);"K";
2020  FOR J=1 TO KJ STEP 2
2030    X$=MID$(KN$,J,2)
2040    IF X$="" THEN X$=MKI$(&H2121)
2050    LPRINT RIGHT$(X$,1);LEFT$(X$,1);
2060  NEXT J
2070  LPRINT CHR$(&H1B);"H";
2080  RETURN
```

〔キーイン後の説明〕

① キーインしたサブルーチンは，明細を印字する部分です。(「1.3.7」MONDAI.K3を参照)

② 変数LCは，印字した明細の件数をカウントするものです。

③ 行番号1130は，明細を20件印字したときに見出し処理を行うようにしています。

設問 8-3 次のサブルーチンをキーインして下さい。

＜キーインするサブルーチン＞

```
3000 ´
3010 *MIDASHI:´ <<< ﾐﾀﾞｼ ｲﾝｼﾞ ｾｲｷﾞｮ ﾙｰﾁﾝ >>>
3020  RESTORE 10000
3030  LPRINT CHR$(&HC);:PC=PC+1              : ´ &hC (ｶｲﾍﾟｰｼﾞ)
3040  LPRINT TAB(69);USING "### ﾍﾟｰｼﾞ";PC
3050  SP=12:BG=1:KJ=34:GOSUB *MID.INJI
3060  LPRINT:LPRINT CHR$(&H1F);CHR$(&H10+2);
3070  SP=0 :KJ=8 :GOSUB *MID.INJI           : ´ SP (ｽﾍﾟｰｽ ﾉ ｶｽﾞ)
3080  SP=24:KJ=10:GOSUB *MID.INJI           : ´ KJ (ｶﾝｼﾞｺｰﾄﾞﾉｶｽﾞ)
3090  SP=24:KJ=14:GOSUB *MID.INJI
3100  LPRINT:LPRINT STRING$(79,"=")
3110  RETURN
4000 *MID.INJI:´ << ﾐﾀﾞｼ ｲﾝｼﾞ ﾙｰﾁﾝ >>
4010  IF SP><0 THEN LPRINT SPC(SP);
4020  LPRINT CHR$(&H1B);"K";                : ´ ESC-K (ｶﾝｼﾞﾓｰﾄﾞ)
4030  IF BG=1 THEN LPRINT CHR$(&HE);        : ´ &HE (ｶｸﾀﾞｲ)
4040  IF UL=1 THEN LPRINT CHR$(&H1B);"X";:  : ´ ESC-X (ｱﾝﾀﾞｰﾗｲﾝ)
4050  FOR J=1 TO KJ
4060    READ X$                             : ´ ｶﾝｼﾞｺｰﾄﾞ ﾖﾐｺﾐ
4070    LPRINT CHR$(VAL("&h"+X$));          : ´ ｶﾝｼﾞ ｲﾝｼﾞ
4080  NEXT J
4090  IF UL=1 THEN LPRINT CHR$(&H1B);"Y";:  : ´ ESC-Y (ｱﾝﾀﾞｰﾗｲﾝ ｶｲｼﾞｮ)
4100  IF BG=1 THEN LPRINT CHR$(&HF);        : ´ &HF (ｶｸﾀﾞｲ ｶｲｼﾞｮ)
4110  LPRINT CHR$(&H1B);"H";                : ´ ESC-H (ｶﾝｼﾞﾓｰﾄﾞ ｶｲｼﾞｮ)
4120  SP=0:BG=0:UL=0:KJ=0
4130  RETURN
10000 DATA 3D,3B,21,21,21,21,3D,6A,21,21,21,21,4F,3F,21,21,21
10010 DATA 21,21,21,30,6C,21,21,21,21,4D,77,21,21,21,21,49,3D
10020 DATA 3B,61,21,21,21,21,4C,3E
10030 DATA 3D,3B,21,21,21,21,21,21,3D,6A
10040 DATA 45,45,21,21,4F,43,21,21,48,56,21,21,39,66
```

〔キーイン後の説明〕

① キーインしたサブルーチンは，見出し印字処理を行う部分です。(「1.3.7」MONDAI.K3を参照)

② 変数PCは，ページ数をカウントするものです。

【 プログラムの確認 】

設 問 8-4 プログラムのリストをプリンタに印字して正しくキーインできているかどうか確認して下さい。

＜確認用リスト＞

```
100 ' ｼﾞｭｳｼｮﾛｸ ｲﾁﾗﾝﾋｮｳ              (MONDAI.8)
110 *HAJIME:' <<< ｼｮｷ ｼｮﾘ >>>
120  DIM INM$(100),INO(100)
130  OPEN "2:JYUSHO.DAT" AS #1
140  FIELD#1,20 AS JNM$,12 AS JYB$,40 AS JJ1$
150  FIELD#1,72 AS DM$,40 AS JJ2$,24 AS JTL$
160  OPEN "2:JYUSHO.IDX" FOR INPUT AS #2
170  FOR I=1 TO 100
180    IF EOF(2) THEN 200 ELSE INPUT#2,INM$(I),INO(I)
190  NEXT I
200 *MAIN:' <<< ﾒｲﾝ ﾙｰﾁﾝ >>>
210  GOSUB *MIDASHI
220  FOR IX=1 TO 100
230    GOSUB *MEISAI
240  NEXT IX
250 *OWARI:' <<< ｼｭｳﾘｮｳ ｼｮﾘ >>>
260  CLOSE
270  END
1000 '
1010 *MEISAI:' <<< ﾒｲｻｲ ｲﾝｼﾞ ｼｮﾘ >>>
1020  IF INM$(IX)="" THEN 1140
1030  GET#1,INO(IX)
1040  KJ=22:KN$=JNM$:GOSUB *KANJI.INJI
1050  LPRINT CHR$(&H1B);"K";CHR$(&H22);CHR$(&H29);CHR$(&H1B);"H";
1060  KJ=14:KN$=JYB$:GOSUB *KANJI.INJI
1070  KJ=42:KN$=JJ1$:GOSUB *KANJI.INJI
1080  KJ=24:KN$=JTL$:GOSUB *KANJI.INJI
1090  LPRINT:LPRINT SPC(30);
1100  KJ=40:KN$=JJ2$:GOSUB *KANJI.INJI
1110  LPRINT:LPRINT
1120  LC=LC+1
1130  IF LC=20 THEN GOSUB *MIDASHI:LC=0
1140  RETURN
2000 *KANJI.INJI:' << ｶﾝｼﾞｲﾝｼﾞ >>
2010  LPRINT CHR$(&H1B);"K";
2020  FOR J=1 TO KJ STEP 2
2030    X$=MID$(KN$,J,2)
2040    IF X$="" THEN X$=MKI$(&H2121)
2050    LPRINT RIGHT$(X$,1);LEFT$(X$,1);
2060  NEXT J
```

(つづく)

```
2070  LPRINT CHR$(&H1B);"H";
2080  RETURN
3000 '
3010 *MIDASHI:' <<< ﾐﾀﾞｼ ｲﾝｼﾞ ｾｲｷﾞｮ ﾙｰﾁﾝ >>>
3020  RESTORE 10000
3030  LPRINT CHR$(&HC);:PC=PC+1              : ' &hC (ｶｲﾍﾟｰｼﾞ)
3040  LPRINT TAB(69);USING "### ﾍﾟｰｼﾞ";PC
3050  SP=12:BG=1:KJ=34:GOSUB *MID.INJI
3060  LPRINT:LPRINT CHR$(&H1F);CHR$(&H10+2);
3070  SP=0 :KJ=8 :GOSUB *MID.INJI          : ' SP (ｽﾍﾟｰｽ ﾉ ｶｽﾞ)
3080  SP=24:KJ=10:GOSUB *MID.INJI          : ' KJ (ｶﾝｼﾞｺｰﾄﾞﾉｶｽﾞ)
3090  SP=24:KJ=14:GOSUB *MID.INJI
3100  LPRINT:LPRINT STRING$(79,"=")
3110  RETURN
4000 *MID.INJI:' << ﾐﾀﾞｼ ｲﾝｼﾞ ﾙｰﾁﾝ >>
4010  IF SP><0 THEN LPRINT SPC(SP);
4020  LPRINT CHR$(&H1B);"K";                : ' ESC-K (ｶﾝｼﾞﾓｰﾄﾞ)
4030  IF BG=1 THEN LPRINT CHR$(&HE);        : ' &HE (ｶｸﾀﾞｲ)
4040  IF UL=1 THEN LPRINT CHR$(&H1B);"X";:  ' ESC-X (ｱﾝﾀﾞｰﾗｲﾝ)
4050  FOR J=1 TO KJ
4060    READ X$                             : ' ｶﾝｼﾞｺｰﾄﾞ ﾖﾐｺﾐ
4070    LPRINT CHR$(VAL("&h"+X$));          : ' ｶﾝｼﾞ ｲﾝｼﾞ
4080  NEXT J
4090  IF UL=1 THEN LPRINT CHR$(&H1B);"Y";: ' ESC-Y (ｱﾝﾀﾞｰﾗｲﾝ ｶｲｼﾞｮ)
4100  IF BG=1 THEN LPRINT CHR$(&HF);        : ' &HF (ｶｸﾀﾞｲ ｶｲｼﾞｮ)
4110  LPRINT CHR$(&H1B);"H";                : ' ESC-H (ｶﾝｼﾞﾓｰﾄﾞ ｶｲｼﾞｮ)
4120  SP=0:BG=0:UL=0:KJ=0
4130  RETURN
10000 DATA 3D,3B,21,21,21,21,3D,6A,21,21,21,21,4F,3F,21,21,21
10010 DATA 21,21,21,30,6C,21,21,21,21,4D,77,21,21,21,21,49,3D
10020 DATA 3B,61,21,21,21,21,4C,3E
10030 DATA 3D,3B,21,21,21,21,21,21,3D,6A
10040 DATA 45,45,21,21,4F,43,21,21,48,56,21,21,39,66
```

【 プログラムの実行 】

設問 8-5 正しくキーインできていたら，プログラムを実行して下さい。

＜実行結果例＞

1 ﾍﾟｰｼﾞ

住　所　録　一　覧　表

氏　名		住　所	電 話 番 号
河合　由絵	〒215	川崎市麻生区高石88 高石荘2－2	
佐藤　純子	〒214	川崎市多摩区生田8－11	044－931－3838
鈴木　宏子	〒168	杉並区宮前4－5－24	03－333－3333
高橋　達彦	〒243－02	厚木市飯山1000	0462－20－0123

（つづく）

田中　豊	〒１５１	渋谷区本町３－４８	０３－３７８－９８７６
原田　一朗	〒２５１	藤沢市藤沢７	０４６６－２３－１１１１
松下　紀子	〒１８６	国立市東３－９	０４２５－７７－７７７７
森川　愛子	〒１８２	調布市柴崎１－１１ 柴崎マンション８０２	０４２４－８８－１２３４
山田　多郎	〒３６３	桶川市鴨川１０－４	０４８７－６３－８０００
渡辺　次郎	〒２１１	川崎市中原区新城３－２６ 第１男子寮	０４４－３８４－５６７８

設問 8-6 このプログラムをドライブ 2 のフロッピーディスクへ「MONDAI.8」という名前で保存して下さい。

＜実行結果例＞

```
save "2:MONDAI.8"
Ok
```

1.6.7 住所録ファイルから宛名書きをするには

市販のDMラベル(タック紙)を使って宛名の印字を行います。

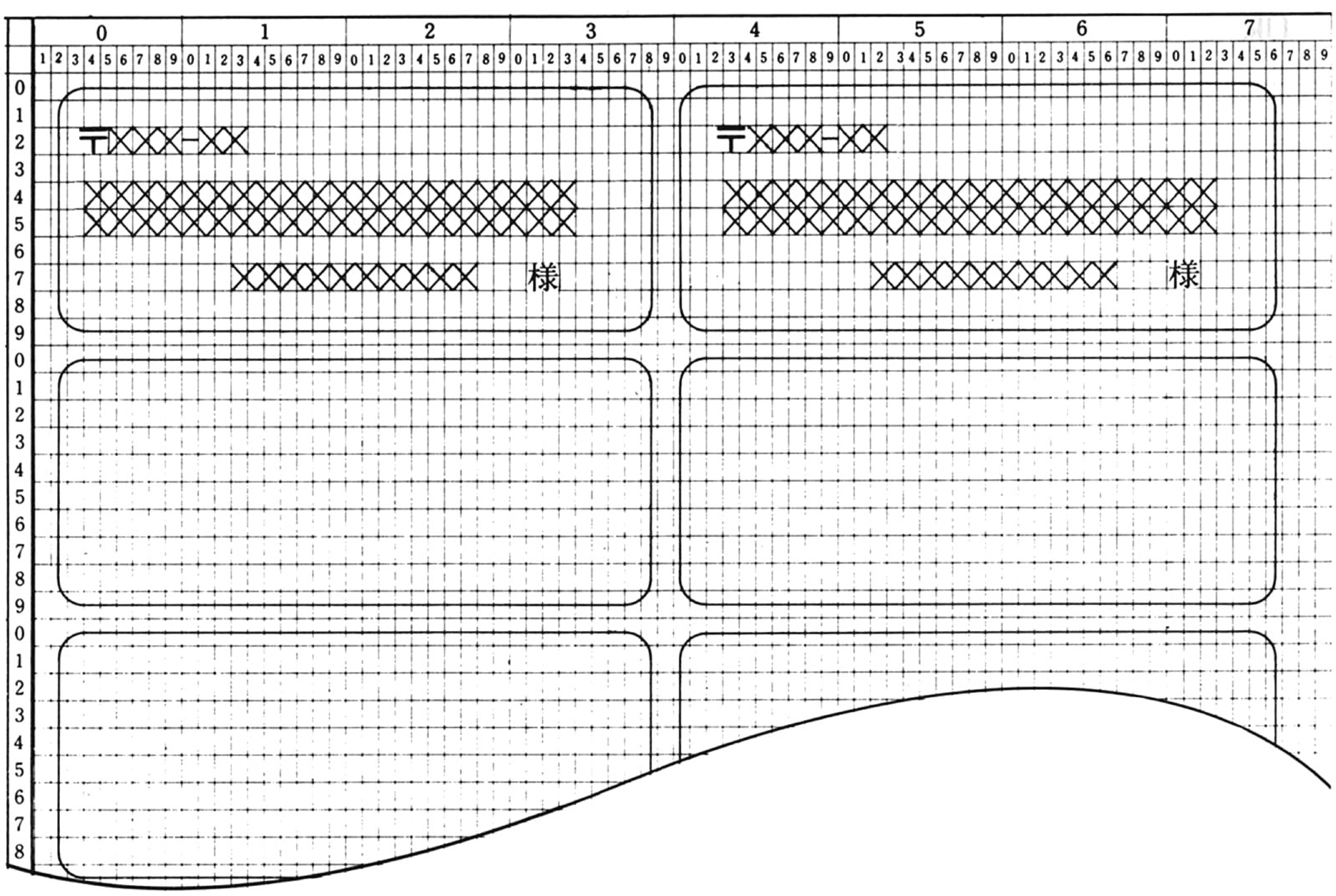
> **問題 9** 索引ファイルを使って住所録データを読み込み,「1.6.1」の帳表フォーマットに従って宛名の印字を行いなさい。

【説 明】

宛名書きを行う場合は，印字する用紙に従ってフォーマットを決めなければなりません。ここでは，一般にタック紙と呼ばれる用紙に印字を行います。汎用紙(ストックフォーム)等に印字を行う場合は，多少用紙がずれても構いませんが，印字位置の限定されているような特別の用紙(伝票等の専用紙やここで使うタック紙等)では，用紙のずれは最小限にとどめなければなりません。

「1.6.1」で示した宛名書き用の帳表フォーマットを次に示しておきます。

<帳表フォーマット>



この設計では,印字を行うときのタックラベルの位置を［　　　］の枠で表しています。したがって実際の印字を行うときは，タックラベルが設計通りになるように用紙のセットを行う必要があります。そこで，用紙をプリンタにセットするときの一応の目安を次に示します。

- 1行目の印字位置は，プリンタの電源を入れたときのプリンタのヘッドの位置になります。したがって，用紙のミシン目をヘッドの上の方に合わせます。(電源を入れた直後に行う)
- 印字の開始位置(レフトマージン)は，PC-8821/PC-8822では目印がありませんが，ヘッドカバーの部分に9本の縦線があります。このうちの一番左の線は，ちょうど2文字目の真ん中になります。これを目印に用紙をセットして下さい。

【 プログラムの作成 】

設問 9-1 いまメモリーに入っているプログラムを利用するために，不要な行を削除して下さい。

<削除する行>

```
210
220
DELETE 1010-1140
DELETE 3000-10040
```

〔削除後の説明〕

① 漢字印字を行うルーチン(＊KANJI.INJI)は，ここでも共通して利用できます。

設問 9-2 さらに次の修正を行って下さい。

<修正>

```
100  ' ｼﾞｭｳｼｮﾛｸ ｲﾁﾗﾝﾋｮｳ          (MONDAI.9)
125  DIM NM$(1),YB$(1),J1$(1),J2$(1),TL$(1)
230  GOSUB *ATENA
240  IF IX<=100 GOTO *MAIN
```

〔修正後の説明〕

① 行番号125は，要素数2個の文字列配列を5個定義しています。これは，印字するタックラベルが左右に2列なので住所録データ2件分を記憶しておくためです。

② 変数IXは，索引テーブルの添字で，行番号240ではこの変数を調べることにより100件分の印字が終わったかどうかを判断しています。

設問 9-3 次のサブルーチンをキーインして下さい。

〈キーインするサブルーチン〉

```
1000 '
1010 *ATENA:' <<< ｱﾃﾅ ｲﾝｼﾞ ｼｮﾘ >>>
1020  GOSUB *DATA.GET
1030  IF I=0 THEN 1190
1040  LPRINT:LPRINT
1050  SP=3 :KJ=16:KN$=MKI$(&H2229)+YB$(0):GOSUB *KANJI.INJI
1060  SP=27:KJ=16:KN$=MKI$(&H2229)+YB$(1):GOSUB *KANJI.INJI
1070  LPRINT:LPRINT
1080  SP=3 :KJ=40:KN$=J1$(0):GOSUB *KANJI.INJI
1090  SP=9 :KJ=40:KN$=J1$(1):GOSUB *KANJI.INJI
1100  LPRINT
1110  SP=3 :KJ=40:KN$=J2$(0):GOSUB *KANJI.INJI
1120  SP=9 :KJ=40:KN$=J2$(1):GOSUB *KANJI.INJI
1130  LPRINT:LPRINT
1140  SP=12:KJ=24:KN$=NM$(0):GOSUB *KANJI.INJI
1150  KJ=4 :KN$=MKI$(&H4D4D):GOSUB *KANJI.INJI
1160  SP=18:KJ=24:KN$=NM$(1):GOSUB *KANJI.INJI
1170  KJ=4 :KN$=MKI$(&H4D4D):GOSUB *KANJI.INJI
1180  LPRINT:LPRINT:LPRINT
1190  RETURN
```

〔キーイン後の説明〕

① 住所録データの各項目を2件分ずつ印字しています。

② 行番号1050～1060は，郵便番号の印字を行います。&H2229は「〒」の漢字コードです。

③ 行番号1080～1090は，住所1の印字を行います。

④ 行番号1110～1120は，住所2の印字を行います。

⑤ 行番号1140～1170は，名前の印字を行います。&H4D4Dは，「様」の漢字コードです。

⑥ 行番号1020は，2件分の住所録データの読み込みと配列へのデータセットを行います。

⑦ 変数Iは，データを入れておく配列の添字ですが，Iが0の場合は印字するデータはありません。

設問 9-4 次のサブルーチンをキーインして下さい。

〈キーインするサブルーチン〉

```
3000 '
3010 *DATA.GET:' << ﾃﾞｰﾀ ﾖﾐｺﾐ >>
3020  FOR I=0 TO 1
3030     IX=IX+1
3040     IF IX<=100 THEN 3100
3050     IF I=0 THEN 3120
```

（つづく）

```
3060     NM$(1)=STRING$(20,&H21)
3070     YB$(1)=STRING$(12,&H21)
3080     J1$(1)=STRING$(40,&H21)
3090     J2$(1)=STRING$(40,&H21)
3100     IF INM$(IX)="" THEN 3030 ELSE GOSUB *DATA.SET
3110   NEXT I
3120   RETURN
4000 *DATA.SET:´ < ﾃﾞｰﾀ ｾｯﾄ >
4010   GET#1,INO(IX)
4020   NM$(I)=JNM$
4030   YB$(I)=JYB$
4040   J1$(I)=JJ1$
4050   J2$(I)=JJ2$
4060   RETURN
```

〔キーイン後の説明〕

① このサブルーチンはデータの読み込みと印字用配列へのセットを行います。

② 変数Iは，印字用配列の添字です。

③ 変数IXは，索引テーブルの添字です。

④ 行番号3060～3090は，最後に印字するデータが1件のときに右側に印字する項目を漢字コードの空白にしておきます。

【 プログラムの確認 】

設問 9-5 プログラムのリストをプリンタに印字して正しくキーインできているかどうか確認して下さい。

＜確認用リスト＞

```
100 ´ ｼﾞｭｳｼｮﾛｸ ｲﾁﾗﾝﾋｮｳ            (MONDAI.9)
110 *HAJIME:´ <<< ｼｮｷ ｼｮﾘ >>>
120  DIM INM$(100),INO(100)
125  DIM NM$(1),YB$(1),J1$(1),J2$(1),TL$(1)
130  OPEN "2:JYUSHO.DAT" AS #1
140  FIELD#1,20 AS JNM$,12 AS JYB$,40 AS JJ1$
150  FIELD#1,72 AS DM$,40 AS JJ2$,24 AS JTL$
160  OPEN "2:JYUSHO.IDX" FOR INPUT AS #2
170  FOR I=1 TO 100
180    IF EOF(2) THEN 200 ELSE INPUT#2,INM$(I),INO(I)
190  NEXT I
200 *MAIN:´ <<< ﾒｲﾝ ﾙｰﾁﾝ >>>
230  GOSUB *ATENA
240  IF IX<=100 GOTO *MAIN
                                                    (つづく)
```

```
250 *OWARI:' <<< ｼｭｳﾘｮｳ ｼｮﾘ >>>
260  CLOSE
270  END
1000 '
1010 *ATENA:' <<< ｱﾃﾅ ｲﾝｼﾞ ｼｮﾘ >>>
1020  GOSUB *DATA.GET
1030  IF I=0 THEN 1190
1040  LPRINT:LPRINT
1050  SP=3 :KJ=16:KN$=MKI$(&H2229)+YB$(0):GOSUB *KANJI.INJI
1060  SP=27:KJ=16:KN$=MKI$(&H2229)+YB$(1):GOSUB *KANJI.INJI
1070  LPRINT:LPRINT
1080  SP=3 :KJ=40:KN$=J1$(0):GOSUB *KANJI.INJI
1090  SP=9 :KJ=40:KN$=J1$(1):GOSUB *KANJI.INJI
1100  LPRINT
1110  SP=3 :KJ=40:KN$=J2$(0):GOSUB *KANJI.INJI
1120  SP=9 :KJ=40:KN$=J2$(1):GOSUB *KANJI.INJI
1130  LPRINT:LPRINT
1140  SP=12:KJ=24:KN$=NM$(0):GOSUB *KANJI.INJI
1150  KJ=4 :KN$=MKI$(&H4D4D):GOSUB *KANJI.INJI
1160  SP=18:KJ=24:KN$=NM$(1):GOSUB *KANJI.INJI
1170  KJ=4 :KN$=MKI$(&H4D4D):GOSUB *KANJI.INJI
1180  LPRINT:LPRINT:LPRINT
1190  RETURN
2000 *KANJI.INJI:' << ｶﾝｼﾞｲﾝｼﾞ >>
2005  IF SP><0 THEN LPRINT SPC(SP);
2010  LPRINT CHR$(&H1B);"K";
2020  FOR J=1 TO KJ STEP 2
2030    X$=MID$(KN$,J,2)
2040    IF X$="" THEN X$=MKI$(&H2121)
2050    LPRINT RIGHT$(X$,1);LEFT$(X$,1);
2060  NEXT J
2070  LPRINT CHR$(&H1B);"H";
2075  SP=0:KJ=0
2080  RETURN
3000 '
3010 *DATA.GET:' << ﾃﾞｰﾀ ﾖﾐｺﾐ >>
3020  FOR I=0 TO 1
3030    IX=IX+1
3040    IF IX<=100 THEN 3100
3050    IF I=0 THEN 3120
3060    NM$(1)=STRING$(20,&H21)
3070    YB$(1)=STRING$(12,&H21)
3080    J1$(1)=STRING$(40,&H21)
3090    J2$(1)=STRING$(40,&H21)
3100    IF INM$(IX)="" THEN 3030 ELSE GOSUB *DATA.SET
3110  NEXT I
3120  RETURN
4000 *DATA.SET:' < ﾃﾞｰﾀ ｾｯﾄ >
4010  GET#1,INO(IX)
4020  NM$(I)=JNM$
4030  YB$(I)=JYB$
4040  J1$(I)=JJ1$
4050  J2$(I)=JJ2$
4060  RETURN
```

【プログラムの実行】

設問 9-6 正しくキーインできていたら，次の手順に従って用紙のセットを行った後，プログラムを実行して宛名の印字を行って下さい。

〔用紙のセット手順〕

① タックラベル用紙をセットします。(4～5cmくらいはみ出すようにしておく)

② プリンタの左上にあるピンチローラーレバーを「PINFEED」にします。

③ タックラベルの左端(用紙の左端ではない)を，ヘッドカバーの上部についている8本の縦線のうち1番左の線に合わせます。

④ ピンチローラーレバーを「FRICTION」にします。

⑤ ヘッドカバーを開けて，用紙のミシン目(上端)がプリンタヘッドの上部(インクリボンの上端)に来るようにプラテンノブを使って用紙を戻します。

⑥ ヘッドカバーを閉じて，SELスイッチを押します。

※ ただし，汎用紙へ印字するときは③～④の操作が不要になります。

<実行結果例>

〒215 川崎市麻生区高石88 高石荘2－2 河合　由絵　様	〒214 川崎市多摩区生田8－11 佐藤　純子　様
〒168 杉並区宮前4－5－24 鈴木　宏子　様	〒243－02 厚木市飯山1000 高橋　達彦　様
〒151 渋谷区本町3－48 田中　豊　様	〒251 藤沢市藤沢7 原田　一朗　様
〒186 国立市東3－9 松下　紀子　様	〒182 調布市柴崎1－11 柴崎マンション802 森川　愛子　様
〒363 桶川市鴨川10－4 山田　多郎　様	〒211 川崎市中原区新城3－26 第1男子寮 渡辺　次郎　様

設問 9-7 このプログラムをドライブ 2 のフロッピーディスクへ「MONDAI.9」という名前で保存して下さい。

<実行結果例>

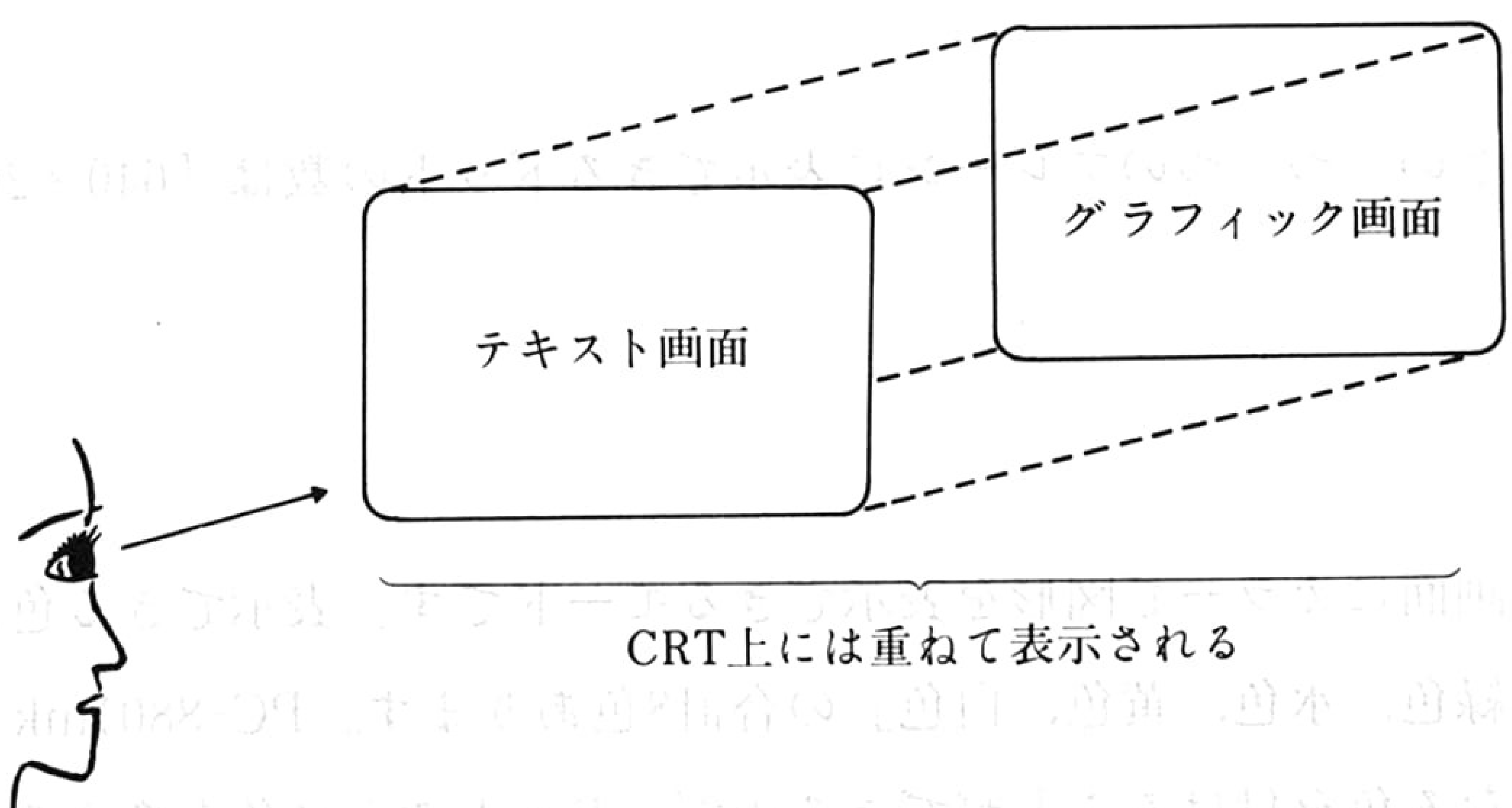
```
save "2:MONDAI.9"
Ok
```

第2章　グラフィックス

この章では，画面にいろいろな図形やグラフを描くための基本的な知識から応用した考え方までを順を追って説明します。ここで取り上げるグラフは，棒グラフ，折れ線グラフ，円グラフ，立体棒グラフ,三次元グラフの5種類です。そのほか図形に模様を付けたり，画面に描いた図形をフロッピーディスクにセーブしたりする方法とタートルを動かす方法についても説明します。

2.1　グラフィック画面について

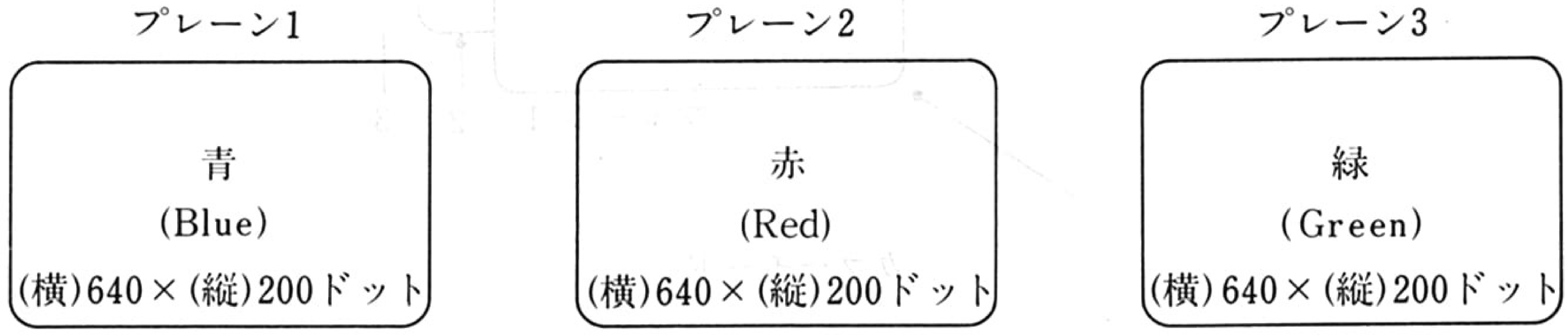
PC-8801mkIIでは，文字を表示する画面と図形やグラフ等を表示する画面が違います。前者をテキスト画面と呼び,後者をグラフィック画面と呼びます。通常，この二つの画面は重なってCRT画面に表示されています。



グラフィック画面はさらにプレーン(またはページ)と呼ばれる3枚の画面に分かれていて,これらの組み合わせでいろいろな使い方ができるようになっています。

2.1.1　プレーンとは

プレーンとは，グラフィック画面を構成する三つの画面のことで，カラーモードではそれぞれ青色(B)，赤色(R)，緑色(G)の光の3原色に割り当てられています。また，おのおののプレーンに表示できる「ドット(点や線を表現する最小の点のこと)」の数や大きさは次のとおりです。



2.1.2 画面モードとは

グラフィック画面には「白黒モード」「カラーモード」および「高分解能白黒モード」と呼ばれる３種類のモードがあり，これらを総称して「画面モード」といいます。これは，3枚のプレーンのおのおのに異なる図形を描いて自由に組み合わせ(重ね合わせ)て使用したり，色の付いた図形を表示するために３枚のプレーンを全部重ね合わせて一つの画面として使用したりするために区別されています。

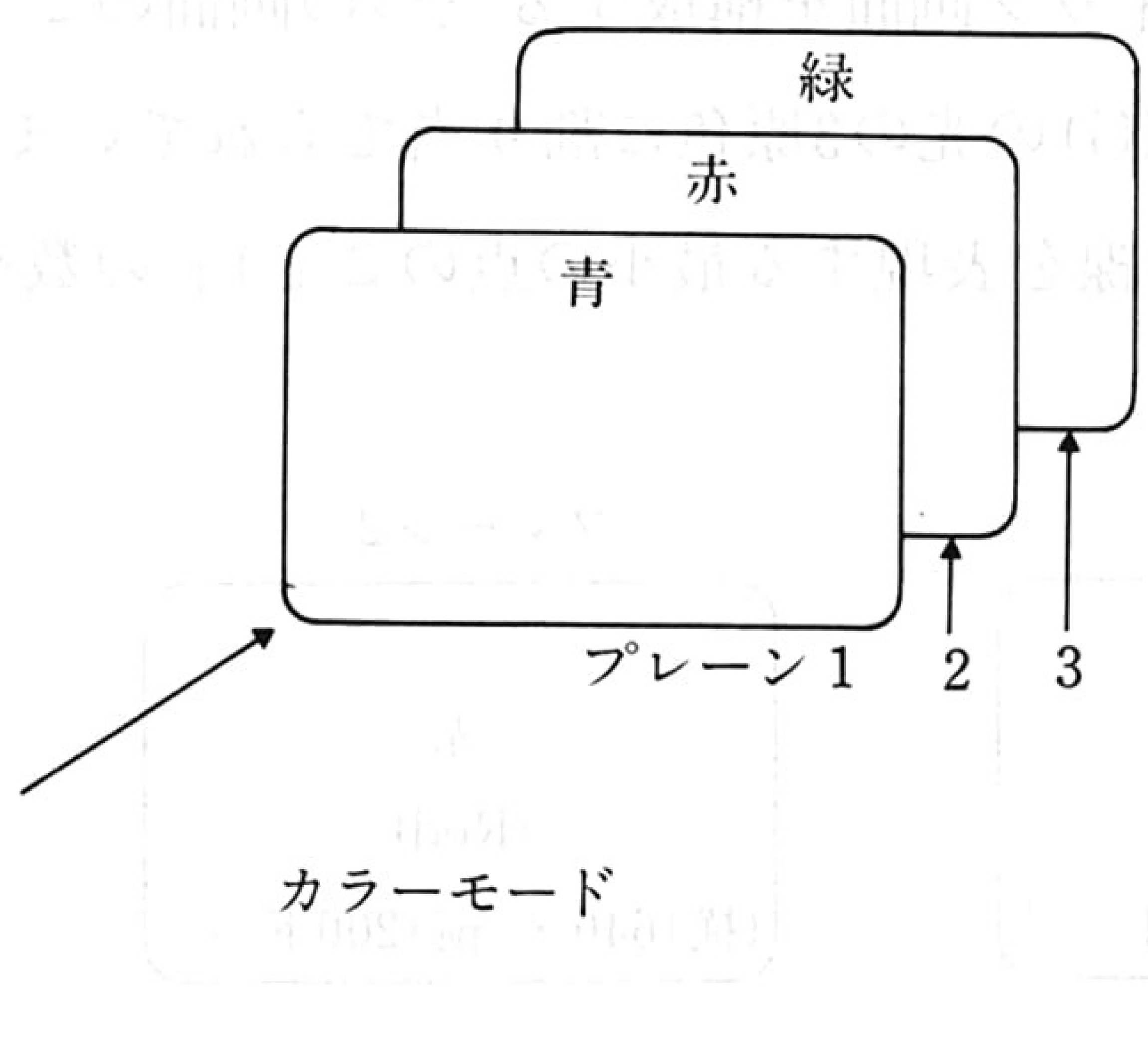
(1)　白黒モード

３枚のプレーンを別々に使用したり，２枚または３枚の別々の図形を重ね合わせて合成した図形を表示するのに使用したりすることができるモードです。たとえば，プレーン１に円，プレーン２に四角形，プレーン３に三角形といった具合に描いておいて，画面への表示はプレーン１(円の表示)だけ，プレーン２(四角形の表示)だけ，プレーン１とプレーン３(円と三角形の表示)の重ね合わせ，プレーン１，プレーン２，プレーン３すべての重ね合わせの表示といったことができます。

なお，このモードのときの一つ一つのプレーンに表示できるドットの数は「640×200」ドットです。

(2)　カラーモード

これは，グラフィック画面にカラーで図形を表示できるモードです。表示できる色は,「黒色，青色，赤色，紫色，緑色，水色，黄色，白色」の合計8色あります。PC-8801mkIIでは，ドットの一つ一つにも異なる色を付けることができるので，ドットごとに色を変えることにより混合色を作ることもできます。また，このモードのときは３枚のプレーンを下の図のように全部重ね合わせて一つの画面として使用しています。そのために白黒モードのように３枚のプレーンを別々に使うことはできません。



図形に付ける色は，その図形を描き込んだプレーンによって変わります。たとえば，青色の四角形といった場合は，プレーン1に四角形が描かれています。そのほか水色の円といった

場合は，プレーン2とプレーン3の同じ位置に同じ大きさの円が描かれています。

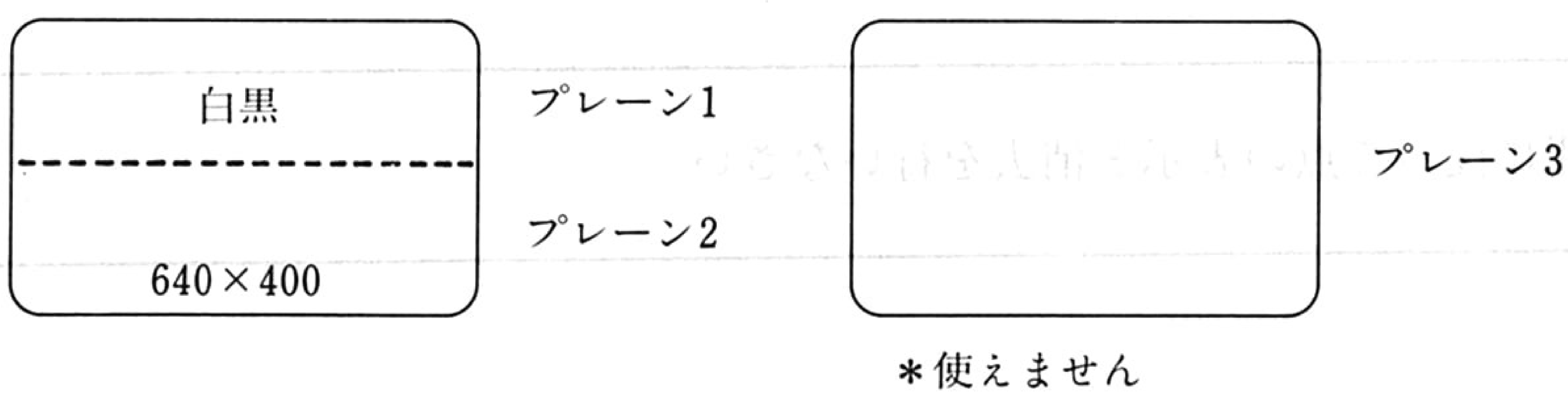
カラーモードでは，図形に付ける色の指定は「パレット番号」で行います。パレットとは絵を描くときに使うパレットと同じ意味で，色を調合する道具といえます。パレット番号についての説明は，「２．２．５　色の変更方法」で行います。ここでは，テキスト画面で文字や数値をカラーで表示する場合のカラーコードと同じものと考えておいて下さい。

なお，このモードで画面に表示するドットの数は「640×200」ドットです。

（３）　高分解能白黒モード

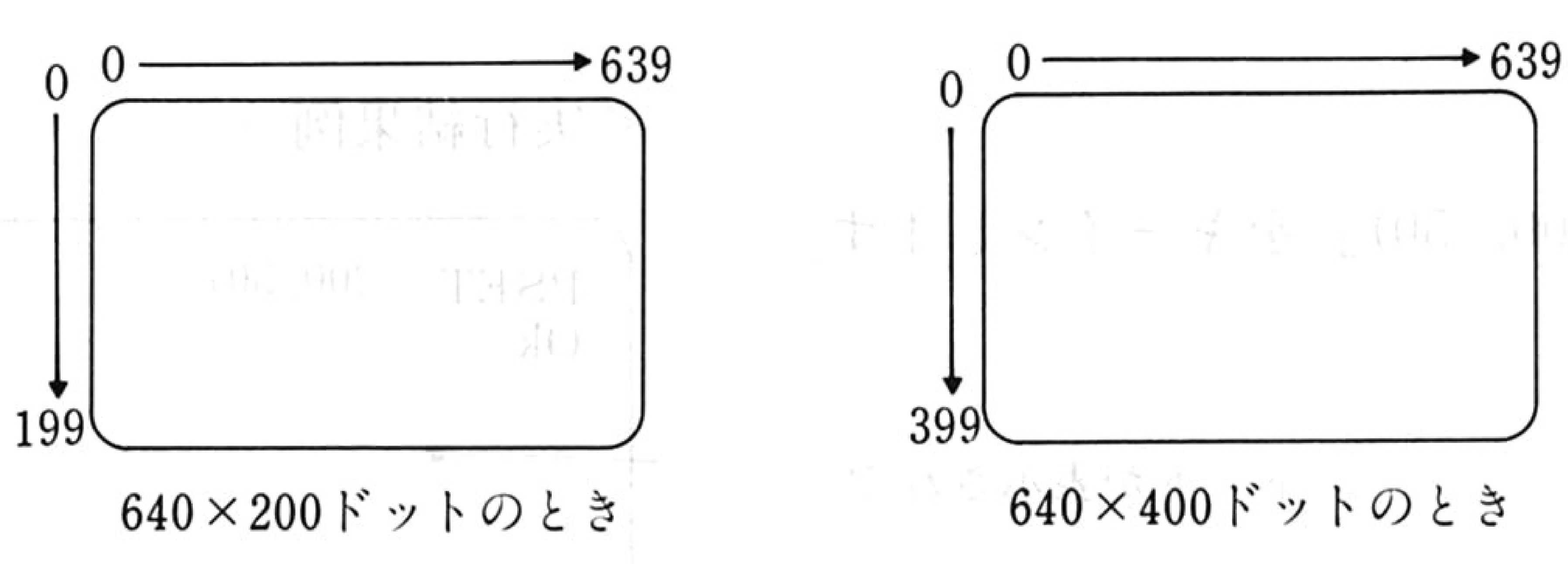
このモードは「640×400」ドットという表示能力を持っています。そのために非常に精密な図形を表示することができますが，白黒モードと同様に白黒でしか表示することができません。

また，次の図のように３枚のプレーンのうち２枚を縦に継ぎ合わせて一つのグラフィック画面として使用するため，３枚目のプレーンは使用できません。



2.1.3　座標について

文字や数値を画面の任意の位置に表示しようとした場合は，LOCATE命令でその位置（「キャラクタ座標」といいます）を指定しました。図形を描く場合もその位置を座標（X，Y）で指定しますが，この場合の座標は「グラフィック座標」と呼ばれキャラクタ座標と違ってドット単位の指定になります。グラフィック座標もキャラクタ座標と同様0から始まります。



2.2 図形を描くための基本命令

ここでは，画面に図形を表示する場合の基本的な命令についてその意味を簡単に説明します。

・PSET／PRESET命令

画面に点(ドット)を表示したり消したりします。

・LINE命令

画面に直線や四角形を描く命令です。

・CIRCLE命令

画面に円を描く命令です。

・COLOR命令

画面に表示する図形の色を決めたり変更したりする命令です。

2.2.1 点の表示と消去方法

問題 10 設問に従って点の表示と消去を行いなさい。

【説　明】

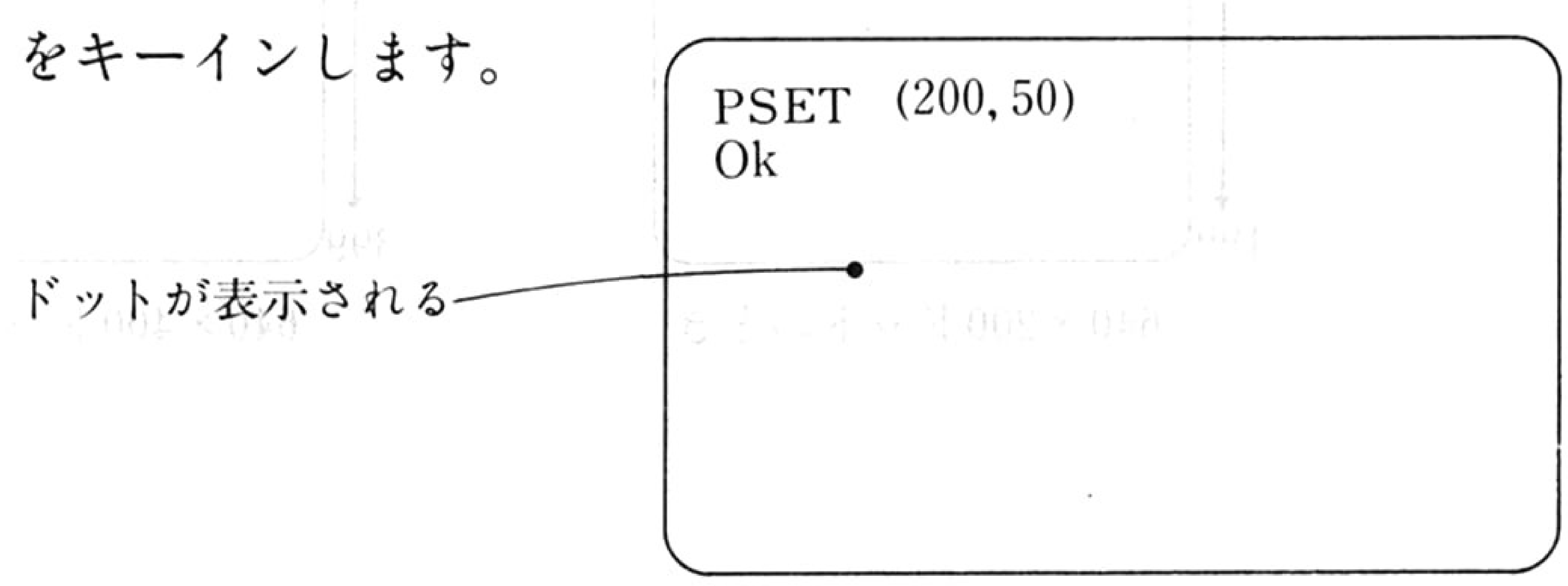
グラフィック画面に1ドットの点を表示したり，表示してある点を消したりするにはPSET／PRESET命令を使います。たとえば，座標(300，100)の位置に点を表示する場合は，「PSET (300，100)」とキーインします。また，この点を消す場合は，「PRESET (300，100)」とキーインします。

設問 10-1 座標(200，50)の位置にドットを表示して下さい。

〔操作説明〕

・命令は「PSET (200，50)」をキーインします。

<実行結果例>



〔実行後の説明〕

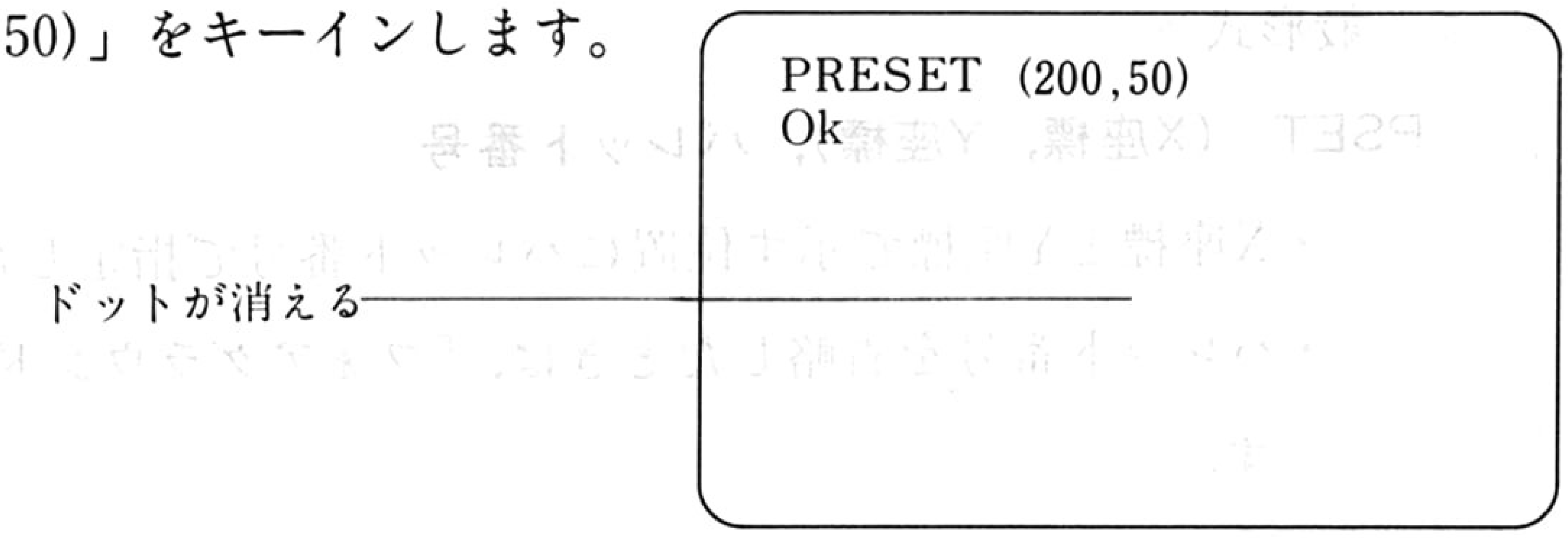
① ここでは命令をメモリーに記憶させるわけではありませんので行番号はいりません。

設問 10-2 結果が正しければ，今度はそのドットを消して下さい。

〔操作説明〕

・命令は「PRESET　(200, 50)」をキーインします。

<実行結果例>

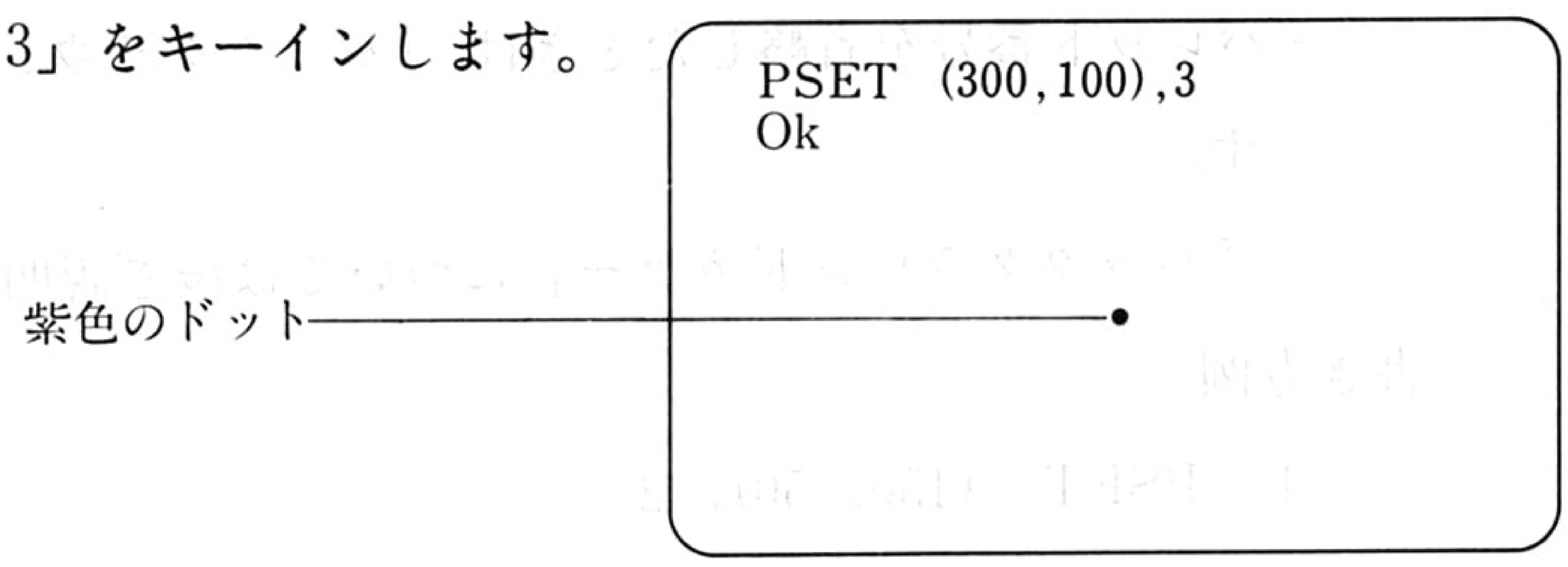


設問 10-3 座標(300, 100)の位置に紫のドットを表示して下さい。

〔操作説明〕

・命令は「PSET　(300, 100),3」をキーインします。

<実行結果例>



〔実行後の説明〕

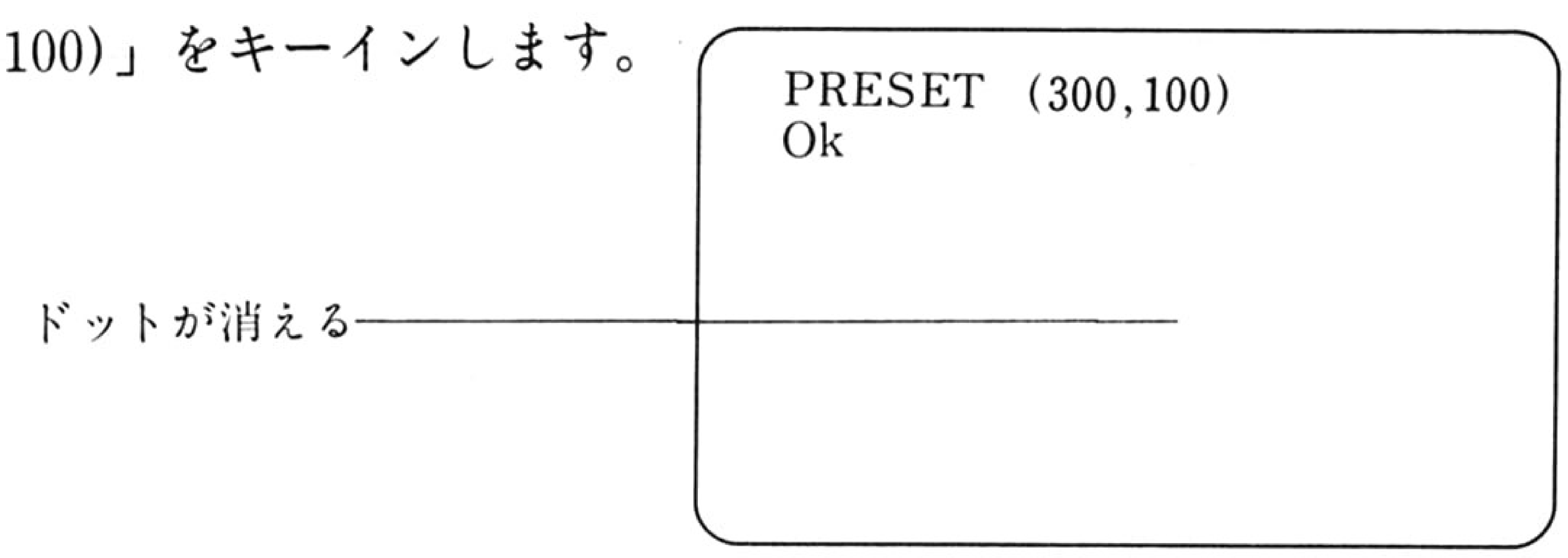
① 表示する点に色を付ける場合は，座標指定の後に続けて表示する色のパレット番号(ここではカラーコードと同じ)を指定するようにします。

設問 10-4 結果が正しければ，今度はそのドットを消して下さい。

〔操作説明〕

・命令は「PRESET　(300, 100)」をキーインします。

<実行結果例>



【 命令のまとめ 】

PSET／PRESET命令の一般形式と書き方例について次に示しておきます。

＜一般形式＞

PSET （X座標，Y座標），パレット番号

・X座標とY座標で示す位置にパレット番号で指定した色の点(ドット)を表示します。

・パレット番号を省略したときは，「フォアグラウンドカラー」の色(通常は白)になります。

「フォアグラウンドカラー」については後で説明します。

PRESET （X座標，Y座標），パレット番号

・X座標とY座標で示す位置に表示されている点をパレット番号で指定した色に変更します。

・パレット番号を省略したときは，「バックグラウンドカラー」の色(通常は黒)になります。

「バックグラウンドカラー」については後で説明します。

＜書き方例＞

① PSET (150, 50), 2

これは，X座標150，Y座標50の位置に赤いドット(点)を表示します。

② PSET (200, 100), 4

これは，X座標200，Y座標100の位置に緑のドットを表示します。

③ PRESET (150, 50)

これは，X座標150，Y座標50の位置に表示されているドットを黒にします。

④ PRESET (200, 100), 2

これはX座標200，Y座標100の位置に表示されているドットの色を赤くします。

2.2.2 線の表示と消去方法

問題 11 設問に従って直線や四角形を描いて下さい。

【説 明】

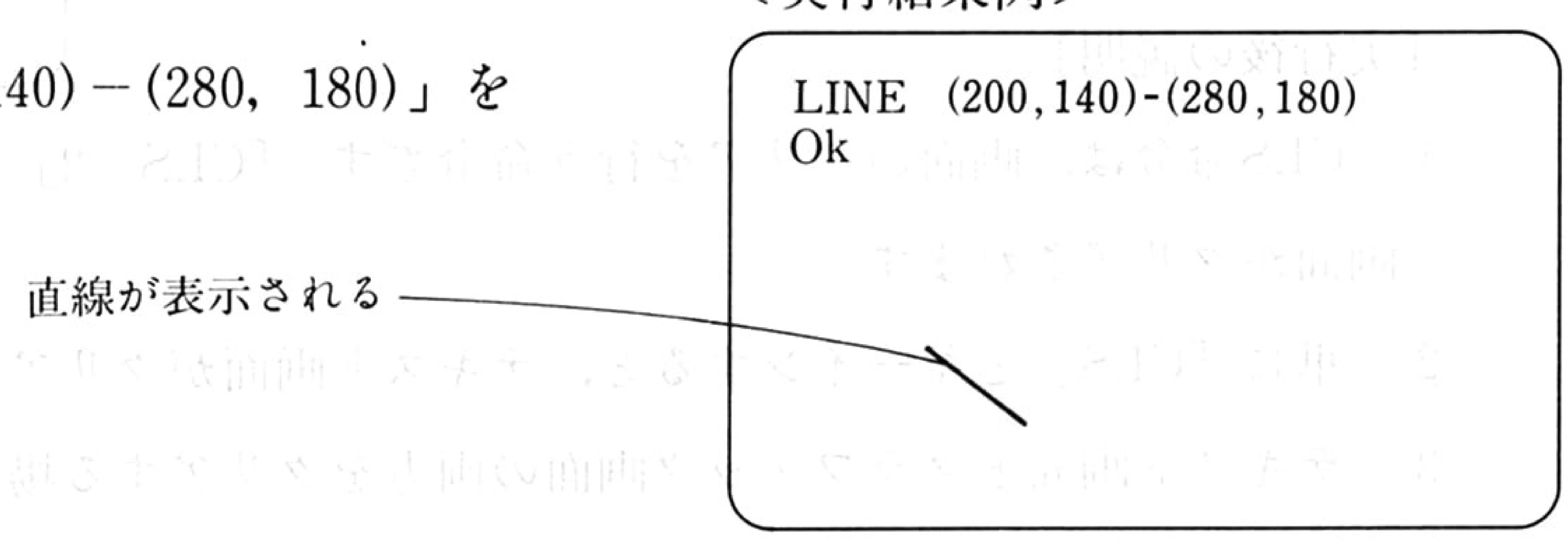
グラフィック画面に線を描いたり四角形を描いたりするにはLINE命令を使います。LINE命令は，2点の座標を指定してそれを結ぶ線を描いたり，その2点を対角線の頂点とする四角形を描いたりします。

設問 11-1 座標(200，140)と座標(280，180)を結ぶ線を描いて下さい。

〔操作説明〕

・命令は「LINE (200，140)－(280，180)」をキーインします。

<実行結果例>

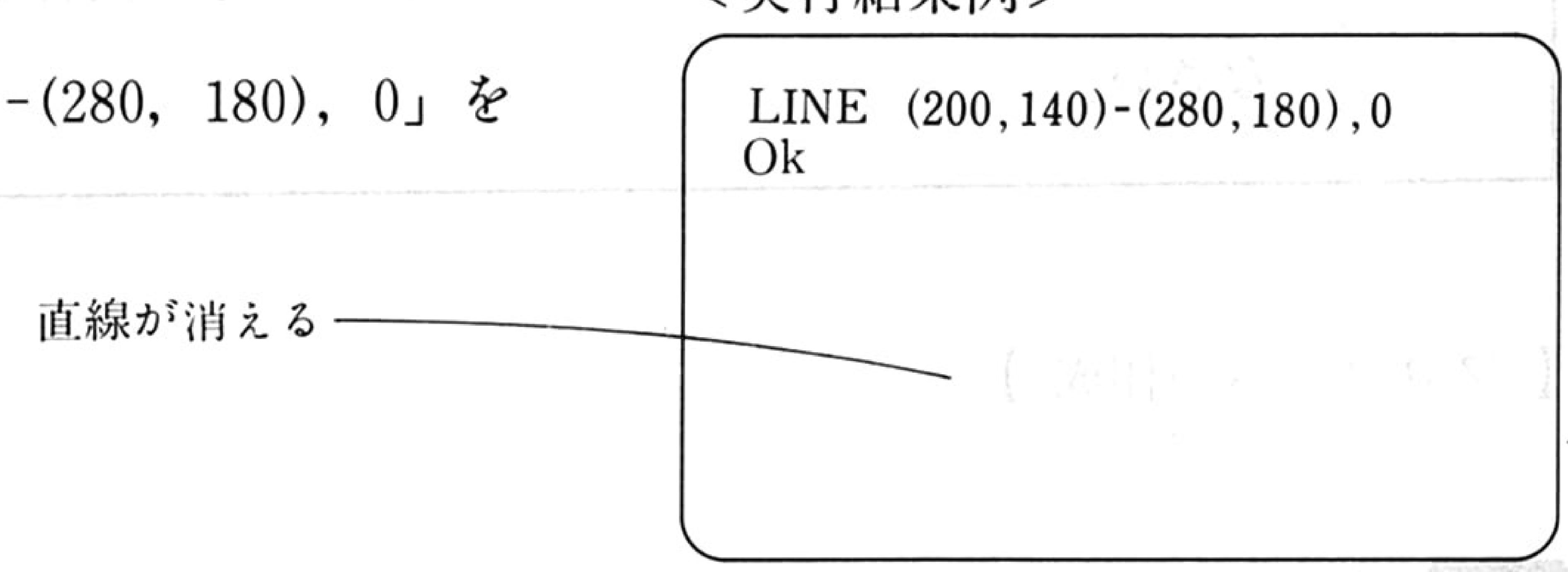


設問 11-2 結果が正しければ，今度はその線を消して下さい。

〔操作説明〕

・命令は「LINE (200，140)-(280，180)，0」をキーインします。

<実行結果例>

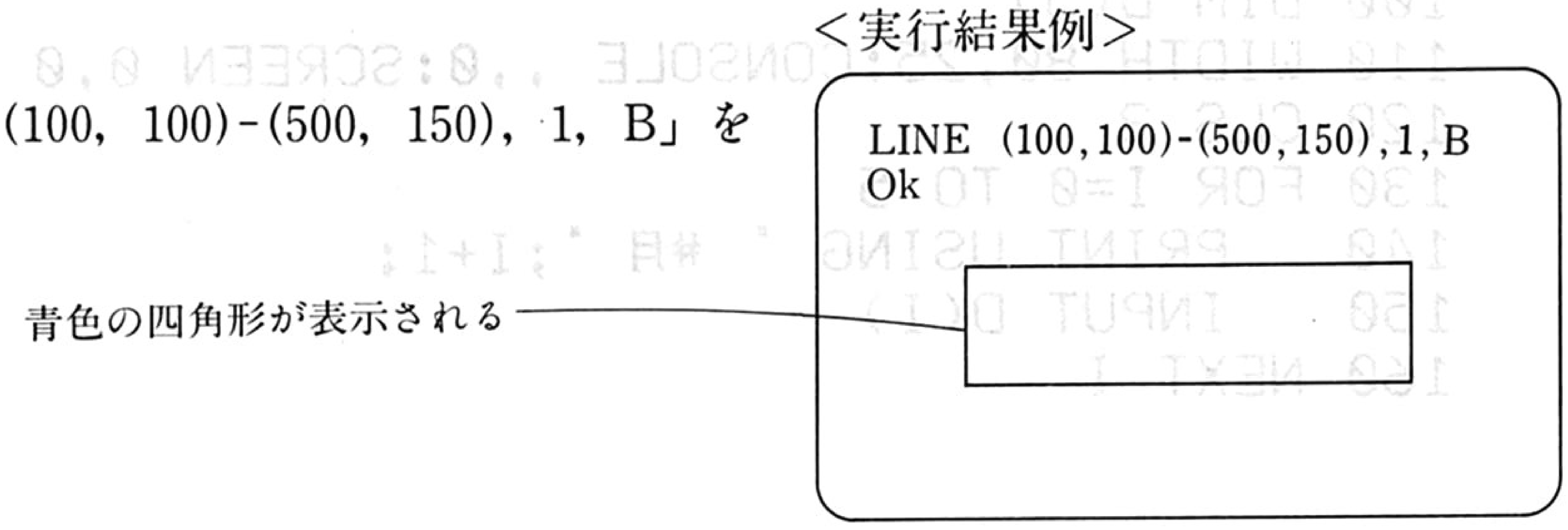


設問 11-3 座標(100，100)と座標(500，150)を対角線の頂点とする青色の四角形を描いて下さい。

〔操作説明〕

・命令は「LINE (100，100)-(500，150)，1，B」をキーインします。

<実行結果例>



〔実行後の説明〕

① 色の指定は，PSET／PRESET命令と同じように座標指定の後に続けてパレット番号を指定します。

② 四角形を表示する場合は，「B(BOX)」を指定します。

③ 四角形の中を塗りつぶす場合は「B」の代わりに「BF」を指定します。

設問 11-4 CLS命令を使って，画面をクリアして下さい。

〔操作説明〕

・命令は「CLS　2」をキーインします。

＜実行結果例＞

```
CLS 2
Ok
```

画面はクリアされる

〔実行後の説明〕

① CLS命令は，画面のクリアを行う命令です。「CLS　2」とキーインするとグラフィック画面がクリアされます。

② 単に「CLS」とキーインすると，テキスト画面がクリアされます。

③ テキスト画面とグラフィック画面の両方をクリアする場合は「CLS　3」とキーインします。

問題　12　半期(6カ月)の売上データを入力して，その割合を棒グラフで示すプログラムを作りなさい。

【 プログラムの作成 】

設問 12-1 メモリーをクリアしてから，次の命令をキーインして下さい。

＜キーインする命令＞

```
100 DIM D(5)
110 WIDTH 80,25:CONSOLE ,,0:SCREEN 0,0
120 CLS 3
130 FOR I=0 TO 5
140   PRINT USING " #月 ";I+1;
150   INPUT D(I)
160 NEXT I
```

〔キーイン後の説明〕

① 行番号100～120は，半期分の売上データを格納しておく配列の確保と画面の設定およびクリアを行う初期処理です。(SCREEN命令については，「2,3 画面モードの設定方法」で説明します)

② 行番号130～160は，半期分の売上データをキーボードから入力して配列D(I)に格納します。

設問 12-2 グラフの枠と目盛りを表示する次の命令を追加して下さい。

<追加する命令>

```
170 LINE (100,180)-(100,0)
180 LINE (100,180)-(570,180)
190 FOR Y=180 TO 0 STEP -18
200   LINE (99,Y)-(101,Y)
210 NEXT Y
```

〔追加後の説明〕

① 行番号170と180は，縦軸と横軸を描きます。

② 行番号190～210は，縦軸の18ドットごとに目盛り線を付けるようにしています。

設問 12-3 棒グラフを描く次の命令を追加して下さい。

<追加する命令>

```
220 FOR I=0 TO 5
230   X=125+72*I
240   LINE (X,180)-(X+50,180-D(I)),I+1,BF
250   X=17+9*I
260   LOCATE X,23:PRINT USING "##月";I+1
270 NEXT I
```

〔追加後の説明〕

① 追加した命令は，入力した半期分の売上データの割合を棒グラフで描くものです。

② 行番号230は，棒グラフの基点となる棒の左下の座標を求めています。

③ 行番号240は，各月の売上データの棒グラフを描きます。

④ 行番号250は，表示した棒グラフの基点となったグラフィック画面のX座標の値から，その棒グラフの下に文字を表示する位置(キャラクタ座標のX座標)を求めています。

⑤ 行番号260は，それぞれの棒グラフの下に何月のものであるかを表示しています。

設問 12-4　終了処理として次の命令を追加して下さい。

＜追加する命令＞

```
280 INPUT "ﾂﾂﾞｹﾏｽｶ (y/n)";YN$
290 IF YN$="y" OR YN$="Y" THEN 120
300 CLS 3:END
```

〔追加後の説明〕

① 終了処理として，プログラムの実行を継続するかどうかを確認するようにしています。

② 処理を継続する場合は，「Y」または「y」を入力することによって行番号120へ戻ります。

【 プログラムの確認 】

設問 12-5　プログラムリストを画面に表示して，正しくキーインできているかどうか確認して下さい。

＜確認用リスト＞

```
100 DIM D(5)
110 WIDTH 80,25:CONSOLE ,,0:SCREEN 0,0
120 CLS 3
130 FOR I=0 TO 5
140   PRINT USING " #月 ";I+1;
150   INPUT D(I)
160 NEXT I
170 LINE (100,180)-(100,0)
180 LINE (100,180)-(570,180)
190 FOR Y=180 TO 0 STEP -18
200   LINE (99,Y)-(101,Y)
210 NEXT Y
220 FOR I=0 TO 5
230   X=125+72*I
240   LINE (X,180)-(X+50,180-D(I)),I+1,BF
250   X=17+9*I
260   LOCATE X,23:PRINT USING "##月";I+1
270 NEXT I
280 INPUT "ﾂﾂﾞｹﾏｽｶ (y/n)";YN$
290 IF YN$="y" OR YN$="Y" THEN 120
300 CLS 3:END
```

【 プログラムの実行 】

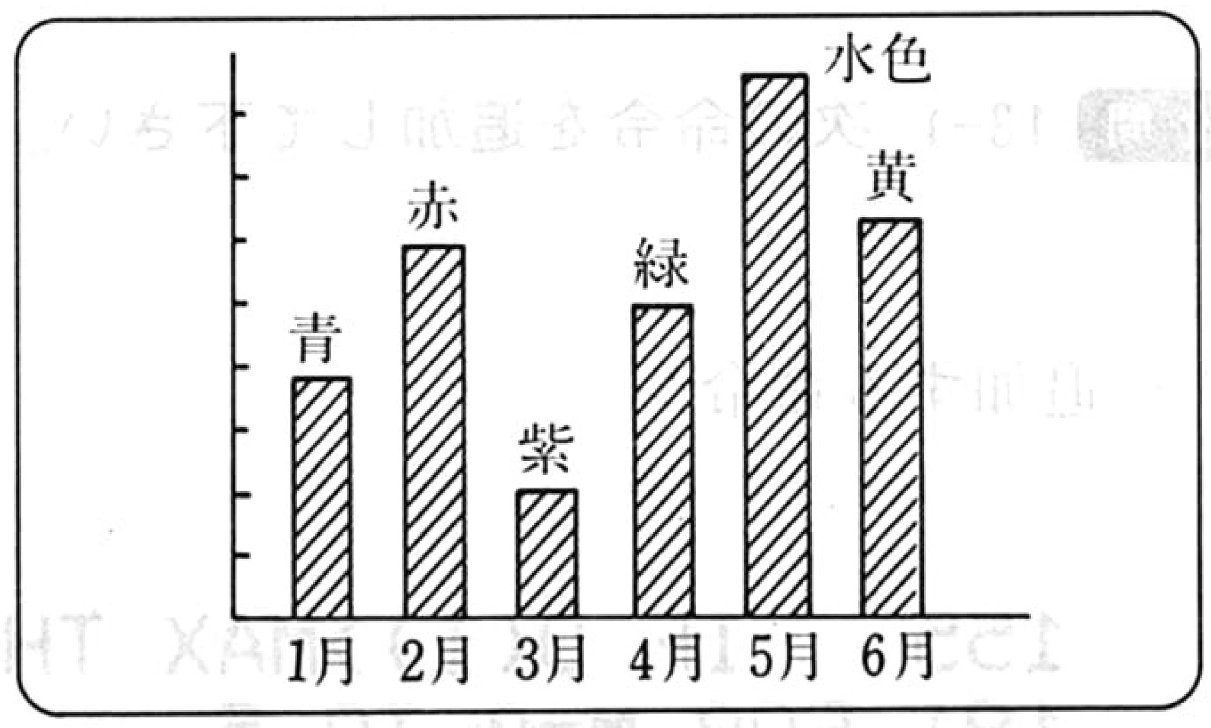
設 問 12-6 正しくキーインできていたら，プログラムを実行して下さい。

〔操作説明〕

入力する売上データは次のとおりです。

1月	70
2月	100
3月	50
4月	90
5月	180
6月	120

＜実行結果例＞



〔実行後の説明〕

① このプログラムでは，入力したデータを棒グラフの高さに使っています。また，このプログラムでは，縦軸の長さを180ドットにしているので，データとして180以上の値を入力しても，棒グラフの高さは180で表されます。

設 問 12-7 このプログラムをドライブ２のフロッピーディスクへ「MONDAI.12」という名前で保存して下さい。

＜実行結果例＞

```
save "2:MONDAI.12"
Ok
```

問題　13　問題12で作ったプログラムを手直しして，折れ線グラフを表示するようにして下さい。

【 プログラムの作成 】

設問 13-1 次の命令を追加して下さい。

＜追加する命令＞

```
155   IF D(I)>MAX THEN MAX=D(I)
181 FOR M=0 TO 5
182   X=100+72*M
183   LINE (X,179)-(X,181)
184   X=X¥8
185   LOCATE X,23:PRINT USING "#月";M+1
186 NEXT M
```

〔追加後の説明〕

① 行番号155は，入力した売上データの中の最大の値を変数MAXに入れるようにしています。

② 行番号181～186は，横軸の72ドットごとに目盛り線を付けその下に月を表示します。

設問 13-2 次のような追加と修正を行って下さい。

＜修正リスト＞

```
215 Y2=D(0)*180/MAX
220 FOR I=1 TO 5
230   Y1=Y2
240   Y2=D(I)*180/MAX
250   X=100+72*(I-1)
260   LINE (X,180-Y1)-(X+72,180-Y2)
```

〔修正後の説明〕

① 変数Y1とY2は，二つの売上データの高さを表しています。

② 行番号215は，最初のデータ(1月の売上)の高さを求めています。

③ 行番号220～260は，2月以降の売上の高さを求めて前の月の売上の高さと直線で結ぶようにしています。

【 プログラムの確認 】

設問 13-3 プログラムのリストをプリンタに印字して，正しくキーインできているかどうか確認して下さい。

＜確認用リスト＞

```
100 DIM D(5)
110 WIDTH 80,25:CONSOLE ,,0:SCREEN 0,0
120 CLS 3
130 FOR I=0 TO 5
140   PRINT USING " #月 ";I+1;
150   INPUT D(I)
155   IF D(I)>MAX THEN MAX=D(I)
160 NEXT I
170 LINE (100,180)-(100,0)
180 LINE (100,180)-(570,180)
181 FOR M=0 TO 5
182   X=100+72*M
183   LINE (X,179)-(X,181)
184   X=X¥8
185   LOCATE X,23:PRINT USING "#月";M+1
186 NEXT M
190 FOR Y=180 TO 0 STEP -18
200   LINE (99,Y)-(101,Y)
210 NEXT Y
215 Y2=D(0)*180/MAX
220 FOR I=1 TO 5
230   Y1=Y2
240   Y2=D(I)*180/MAX
250   X=100+72*(I-1)
260   LINE (X,180-Y1)-(X+72,180-Y2)
270 NEXT I
280 INPUT "ﾂﾂﾞｹﾏｽｶ (y/n)";YN$
290 IF YN$="y" OR YN$="Y" THEN 120
300 CLS 3:END
```

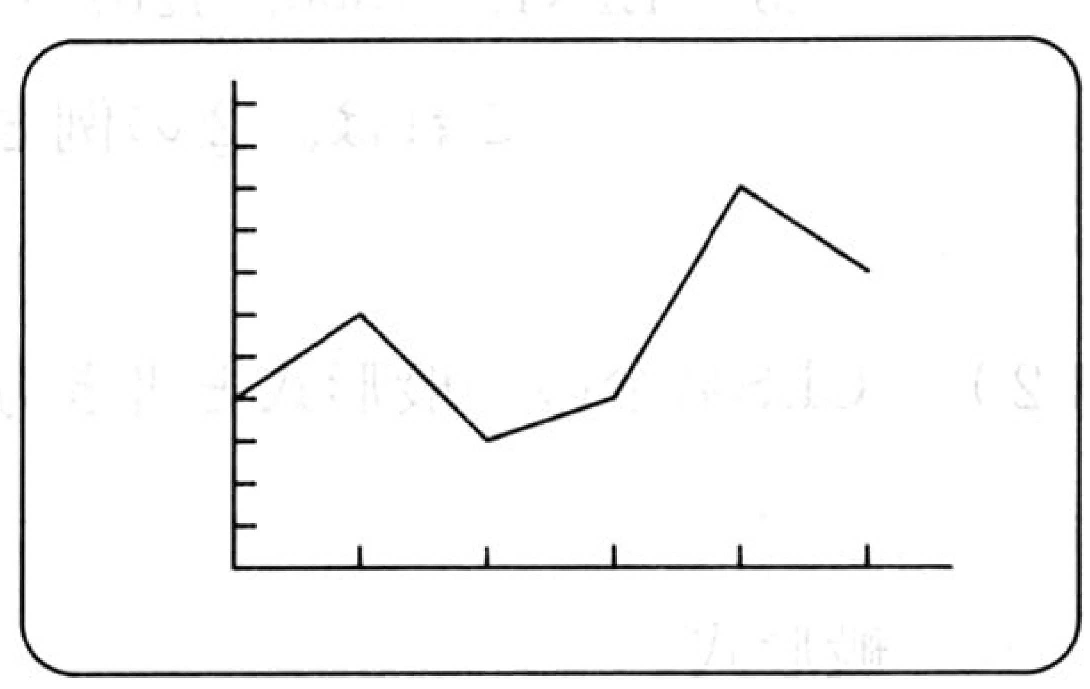
【 プログラムの実行 】

設問 13-4 正しくキーインできていたら，プログラムを実行して下さい。

〔操作説明〕

入力する売上データは，設問12-6で入力したものと同じです。

＜実行結果例＞



〔実行後の説明〕

① ここでは，問題12での棒グラフの表示の仕方とは違って，入力したデータの最大の値を上限値として各月の割合を示すようにしています。

② 入力する売上データの値は，180以上でも構いません。

設問 13-5 このプログラムをドライブ2のフロッピーディスクへ「MONDAI.13」という名前で保存して下さい。

<実行結果例>

```
save "2:MONDAI.13"
Ok
```

【 命令のまとめ 】

（1） LINE命令の一般形式と書き方例を次に示します。

<一般形式>

LINE （X_1座標，Y_1座標）-（X_2座標，Y_2座標），パレット番号，BまたはBF

・座標（X_1，Y_1）の位置と座標（X_2，Y_2）の位置を結ぶ直線または四角い枠をパレット番号で指定した色で表示します。

・「B」を指定すると（X_1，Y_1）の位置と（X_2，Y_2）の位置を対角線とする四角い枠をパレット番号で指定した色で表示します。

・「BF」を指定すると四角い枠の内側も更にパレット番号で指定した色で塗られます。

<書き方例>

① LINE （100，50）-（200，100），4

これは，座標（100，50）の位置と座標（200，100）の位置を結ぶ緑色の直線を表示します。

② LINE （300，50）-（400，100），4，B

これは，座標（300，50）の位置と座標（400，100）の位置を対角線とする緑色の四角い枠を表示します。

③ LINE （300，120）-（400，170），4，BF

これは，②の例と同じことを行った後，四角い枠の内側も緑色に塗ります。

（2） CLS命令の一般形式と書き方例を次に示します。

<一般形式>

CLS 機能コード

・機能コードで指定した画面をクリアします。

・機能コードには次の種類があります。

1 ： テキスト画面だけをクリアします。

2 ： グラフィック画面だけをクリアします。

3 ： テキスト画面とグラフィック画面の両方をクリアします。

・機能コードを省略するとテキスト画面だけをクリアします。

＜書き方例＞

① CLS 3

これは，テキスト画面とグラフィック画面の両方をクリアします。

2.2.3 円の表示と消去方法

問題 14 設問に従っていろいろな円を描いて下さい。

【説 明】

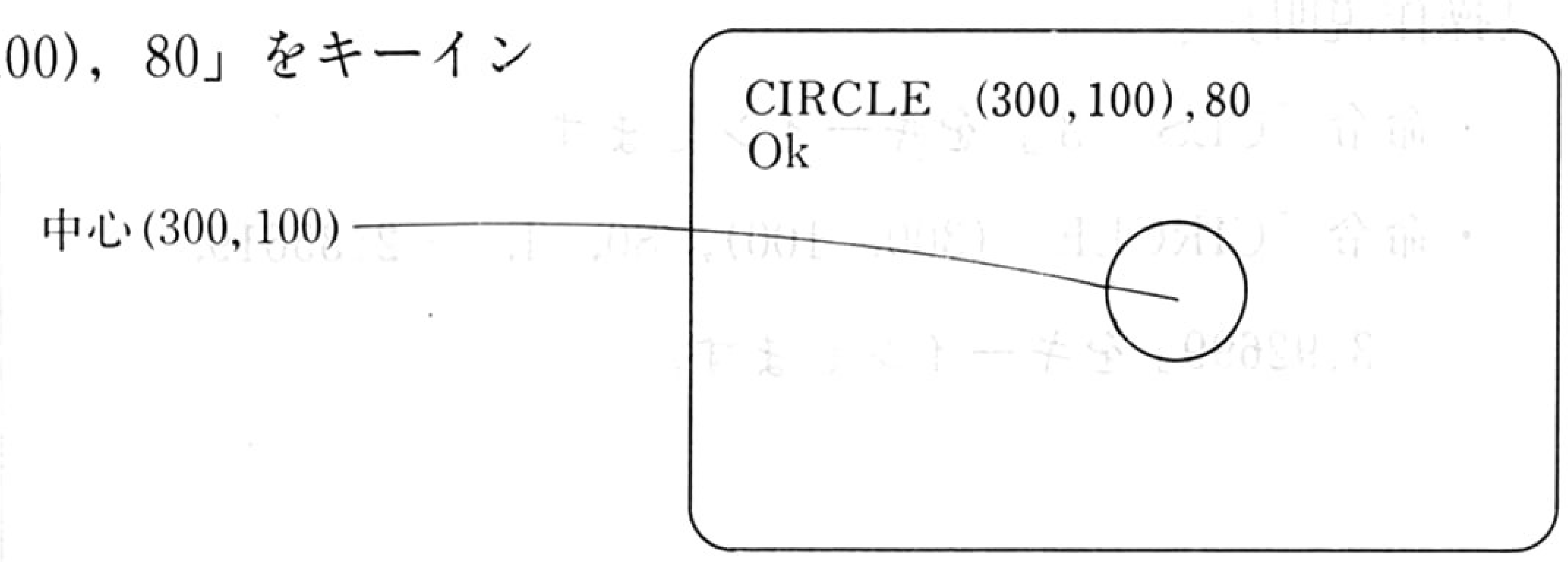
円を描くには「CIRCLE」という命令を使います。CIRCLE命令は，中心となる座標と半径を指定することによって円を描くことができますが，このほかに開始の角度と終了の角度を指定することによって扇型を描いたり円弧を描くことができます。また，縦方向と横方向の半径の比率を変更することによって，縦長や横長の楕円を描くこともできます。

設問 14-1 座標(300，100)の位置を中心とする半径80の正円を表示して下さい。

〔操作説明〕

・命令は「CIRCLE (300, 100), 80」をキーインします。

＜実行結果例＞



〔実行後の説明〕

① 中心となる座標を指定した後，半径を指定します。

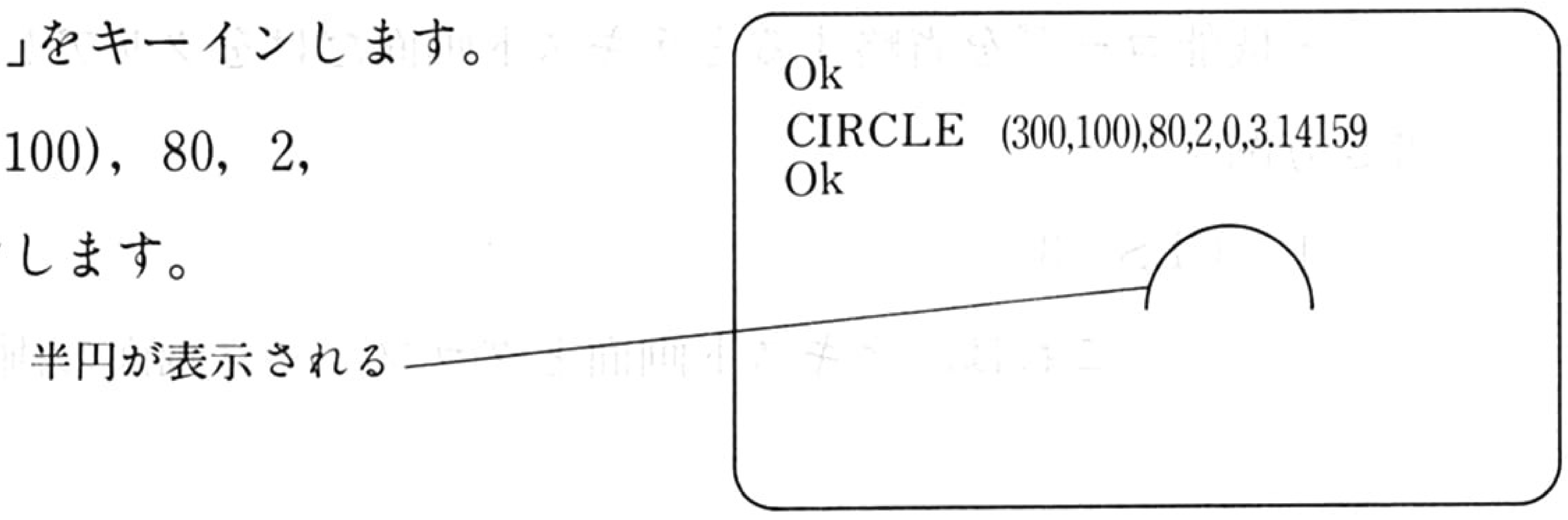
設問 14-2 グラフィック画面とテキスト画面の両方をクリアしてから，座標（300，100）を中心とする半径80の上半分（角度０～π）の円弧を赤色で表示して下さい。

〔操作説明〕

・命令は最初に「CLS　3」をキーインします。

・次に「CIRCLE　(300，100)，80，2，0，3.14159」をキーインします。

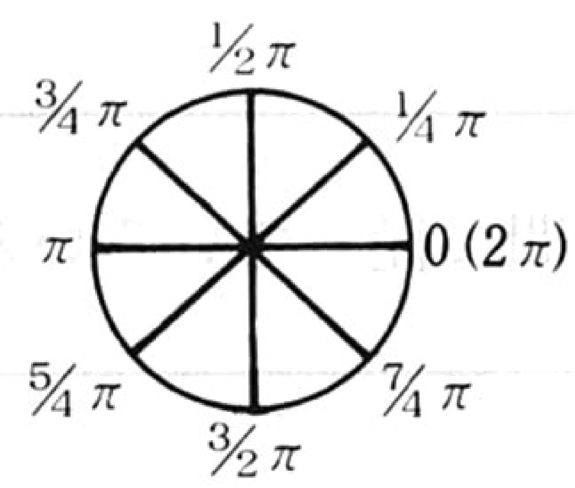
＜実行結果例＞



〔実行後の説明〕

① カラーで表示する場合は，パレット番号を半径に続けて指定します。

② 半径などを表示する場合は，表示する円弧の開始角度と終了角度をラジアンで指定します。ここでは，0～πの円弧を表示しています。角度の指定は右のとおりです。



π＝3.14159

設問 14-3 グラフィック画面とテキスト画面の両方をクリアしてから，次の条件で緑色の扇形の図形を表示して下さい。

＜条件＞

① 中心の座標を(300，100)とします。

② 半径を「80」とします。

③ 開始角度を「¾π」，終了角度を「5/4π」とします。

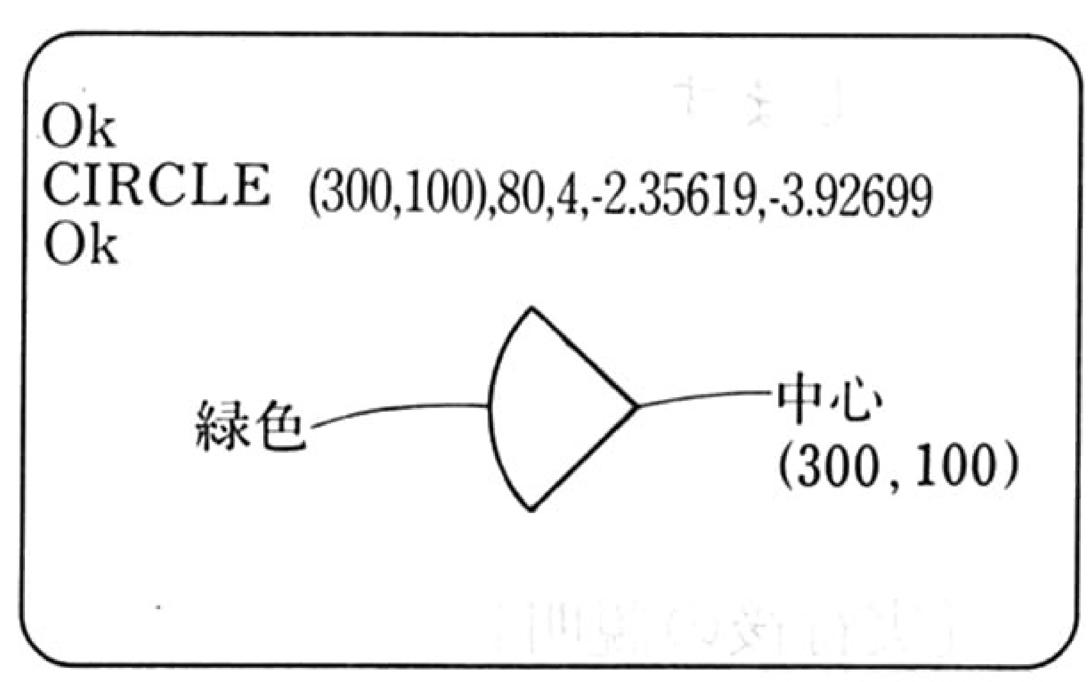
④ 命令は「CIRCLE　(300，100)，80，4，−2.35619，−3.92699」をキーインします。

〔操作説明〕

・命令「CLS　3」をキーインします。

・命令「CIRCLE　(300，100)，80，4，−2.35619，−3.92699」をキーインします。

＜実行結果例＞



〔実行後の説明〕

① 開始角度または終了角度に「−」を付けると，円周上の点と中心とを結ぶ直線を引くことができます。（扇形の表示）

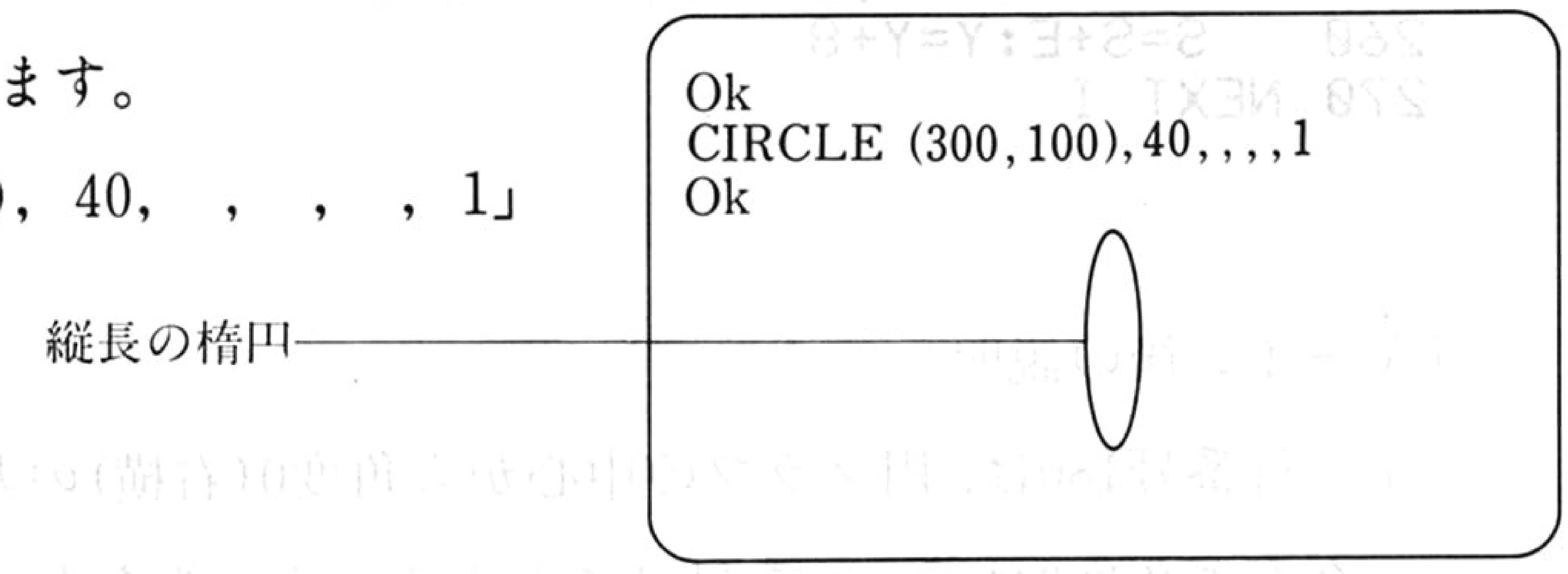
設問 **14-4　グラフィック画面とテキスト画面の両方をクリアしてから，座標(300，100)を中心とする縦の半径80，横の半径40の縦長の楕円を表示して下さい。**

〔操作説明〕

・命令「CLS　3」をキーインします。

・命令は「CIRCLE　(300，100)，40，　，　，　，1」をキーインします。

<実行結果例>



〔実行後の説明〕

① 横軸と縦軸の半径の比率を自由に設定することができます。これによって，縦長や横長の楕円を自由に表示することができます。(ただし，画面が640×200の場合「比率＝縦軸半径／横軸半径＊0.5」になります)

問題　15　問題12で作ったプログラムを手直しして，半期分の売上の比を円グラフで表示するプログラムを作りなさい。

【 プログラムの作成 】

設問 **15-1　行番号155を次のように修正して下さい。**

＜修正リスト＞

```
155     G=G+D(I)
```

〔修正後の説明〕

① 円グラフは，全体に対する各データの割合を示すものです。したがって，全データの合計をとるようにします。

設問 **15-2　行番号170～300の命令を削除した後，次の命令をキーインして下さい。**

＜キーインする命令＞

```
170 S=0:Y=40
180 LINE (200,100)-(350,100),1
190 FOR I=0 TO 5
200   E=D(I)*6.28318/G
```

（つづく）

```
210    IF E=0 THEN 270
220    CIRCLE (200,100),150,I+1,-S,-(S+E)
230    LINE (384,Y)-STEP(32,7),I+1,B
240    P=D(I)*100/G
250    LOCATE 55,I+5:PRINT USING "#月 = ### --> ###.#%";I+1;D(I);P
260    S=S+E:Y=Y+8
270 NEXT I
```

〔キーイン後の説明〕

① 行番号180は，円グラフの中心から角度0(右横)の方向に線を引きます。これは，CIRCLE命令で角度0に「−」を付けても中心からの線を引くことができないからです。

② 行番号190～270は，各月の売上の円グラフを表示します。

③ 行番号200は，全体に対する各月の売上の割合を角度で求めています。

④ 行番号210は，売上の割合が0のとき，次のデータへ行くようにしています。

⑤ 行番号220は，求めた角度に従って円グラフを表示しています。

⑥ 行番号230～260は，どの色が何月のデータで何%であるかを円グラフの右横に表示しています。

【 プログラムの確認 】

設問 15-3 プログラムのリストをプリンタに印字して，正しくキーインできているかどうか確認して下さい。

<確認用リスト>

```
100 DIM D(5)
110 WIDTH 80,25:CONSOLE ,,0:SCREEN 0,0
120 CLS 3
130 FOR I=0 TO 5
140    PRINT USING " #月 ";I+1;
150    INPUT D(I)
155    G=G+D(I)
160 NEXT I
170 S=0:Y=40
180 LINE (200,100)-(350,100),1
190 FOR I=0 TO 5
200    E=D(I)*6.28318/G
210    IF E=0 THEN 270
220    CIRCLE (200,100),150,I+1,-S,-(S+E)
230    LINE (384,Y)-STEP(32,7),I+1,B
240    P=D(I)*100/G
250    LOCATE 55,I+5:PRINT USING "#月 = ### --> ###.#%";I+1;D(I);P
260    S=S+E:Y=Y+8
270 NEXT I
```

【プログラムの実行】

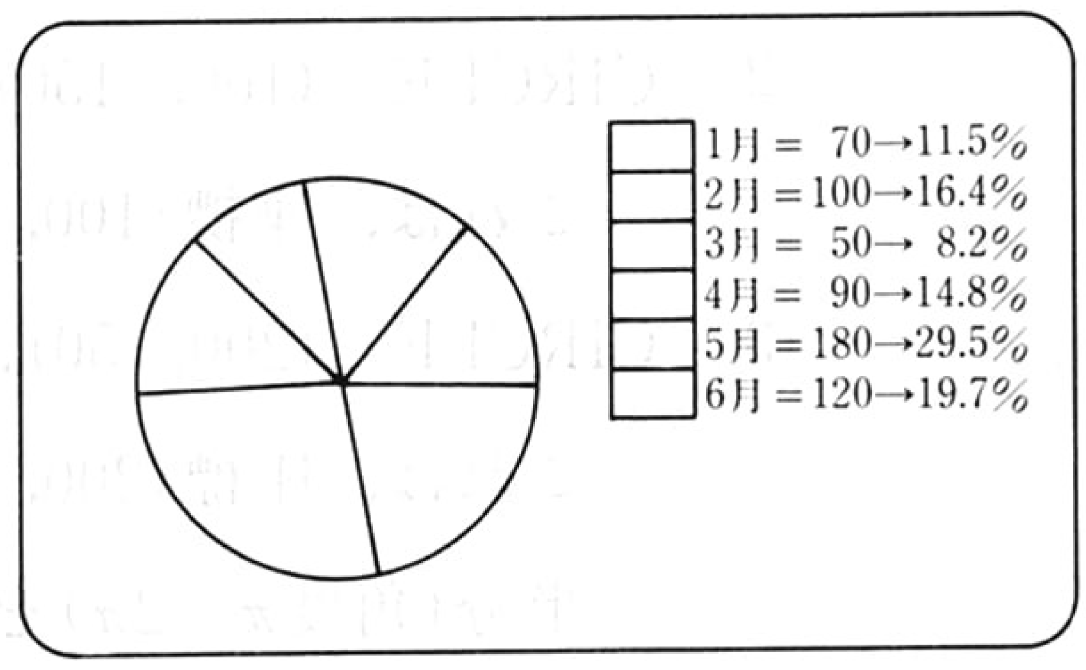
設問 15-4 正しくキーインできていたら，プログラムを実行して下さい。

〔操作説明〕

入力する売上データは次のとおりです。

1月	70
2月	100
3月	50
4月	90
5月	180
6月	120

＜実行結果例＞



〔実行後の説明〕

① ここで表示する円グラフは，角度0π(右横)の位置から左回りに表示していきます。

【命令のまとめ】

CIRCLE命令の一般形式と書き方例を次に示します。

＜一般形式＞

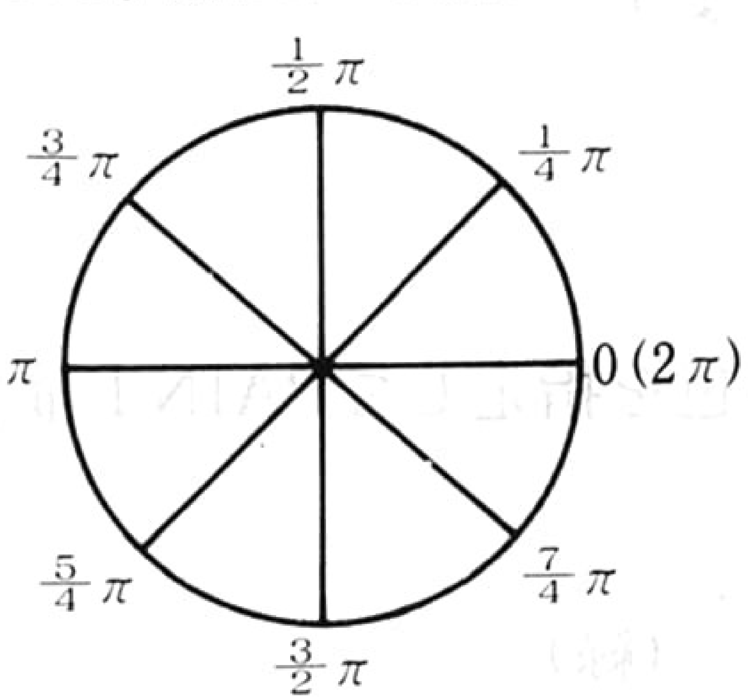
CIRCLE （X座標，Y座標)，半径，パレット番号，開始角度，終了角度，比率

・座標(X，Y)の位置を中心にして指定した半径の円,円弧,扇形,楕円などを表示します。

・半径は必ず指定しなければなりません。

・円をカラーで表示するときは，パレット番号で色を指定します。

・円弧の図形を表示するときは，開始角度と終了角度を「0～2π」の範囲で指定しなければなりません。(指定は実数で行います)



$\pi=3.14159$

・開始角度または終了角度に「－」の符号を付けると，符号を付けた方の円周上の点と中心点とを結ぶ直線を表示します。(角度0°のときは直線が表示されないので，直線を表示する場合は「－2π」を指定します)

・開始角度および終了角度を省略すると正円または楕円が表示されます。

・楕円を表示するときは，縦軸の半径と横軸の半径の比率を「(縦軸の半径)／(横軸の半径)」の値で指定します。

<書き方例>

① CIRCLE (100, 50), 40

これは，座標(100, 50)の位置を中心とする半径40の白い正円を表示します。

② CIRCLE (100, 150), 40, 1

これは，座標(100, 150)の位置を中心とする半径40の青い正円を表示します。

③ CIRCLE (200, 50), 40, 4, 3.14159, 6.28318, 2

これは，座標(200, 50)の位置を中心とする半径40の縦横比率2の緑色の楕円の下半分(角度π～2π)を表示します。

④ CIRCLE (300, 100), 40, 6, −0.78540, −2.35619

これは，座標(300, 100)の位置を中心とする半径40の黄色の扇形(角度$\frac{1}{4}\pi$ ～$\frac{3}{4}\pi$)の図形を表示します。

2.2.4 色を塗る方法

問題 16 問題15で作成した円グラフに色を塗って下さい。

【説 明】

表示されている図形に色を塗るには「PAINT」という命令を使います。この命令は，指定した色の線で囲まれる図形の内側を指定した色で塗りつぶすものです。たとえば，緑色の線で表示された円の内側を青色で塗る場合は，次のように行います。

① 緑色の円を表示する

CIRCLE (300, 100), 60, 4

② 円の内側の座標と塗りつぶす色および枠の色を指定してPAINT命令を実行します。

PAINT (300, 100), 1, 4

- 4 … 枠の色 (緑)
- 1 … 塗りつぶす色 (青)
- (300, 100) … 円の内側の座標であればどこでも構いません。

設問 16-1 いまメモリーに記憶しているプログラムを表示して，次のような修正と追加を行って下さい。

<修正と追加>

```
180 LINE (200,100)-(350,100),7
220   CIRCLE (200,100),150,7,-S,-(S+E)
222   XX=200+147*COS(S+E/2)
224   YY=100-147*SIN(S+E/2)*.41
226   PAINT (XX,YY),I+1,7
230   LINE (384,Y)-STEP(32,7),7,B
235   PAINT (400,Y+1),I+1,7
```

〔修正後の説明〕

① 行番号222と224は，円の内側になる位置の座標を求めています。

② 行番号226は，白色で囲まれた図形の内側を指定の色で塗る命令です。

【 プログラムの確認 】

設問 16-2 プログラムのリストをプリンタに印字して，正しくキーインできているかどうか確認して下さい。

<確認用リスト>

```
100 DIM D(5)
110 WIDTH 80,25:CONSOLE ,,0:SCREEN 0,0
120 CLS 3
130 FOR I=0 TO 5
140   PRINT USING " #月 ";I+1;
150   INPUT D(I)
155   G=G+D(I)
160 NEXT I
170 S=0:Y=40
180 LINE (200,100)-(350,100),7
190 FOR I=0 TO 5
200   E=D(I)*6.28318/G
210   IF E=0 THEN 270
220   CIRCLE (200,100),150,7,-S,-(S+E)
222   XX=200+147*COS(S+E/2)
224   YY=100-147*SIN(S+E/2)*.41
226   PAINT (XX,YY),I+1,7
230   LINE (384,Y)-STEP(32,7),7,B
235   PAINT (400,Y+1),I+1,7
240   P=D(I)*100/G
250   LOCATE 55,I+5:PRINT USING "#月 = ### --> ###.#%";I+1;D(I);P
260   S=S+E:Y=Y+8
270 NEXT I
```

【 プログラムの実行 】

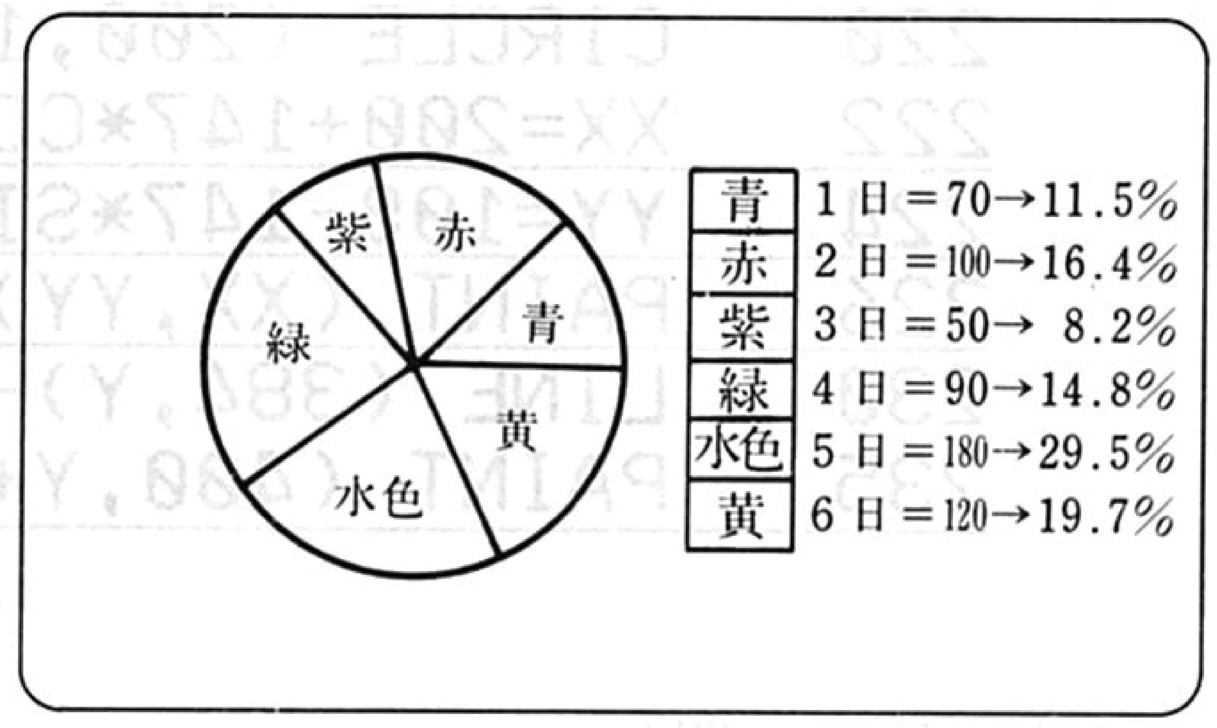
設問 16-3 正しくキーインができていたら，プログラムを実行して下さい。

〔操作説明〕

入力した売上データは次のとおりです。

1月	70
2月	100
3月	50
4月	90
5月	180
6月	120

<実行結果例>



設問 16-4 このプログラムをドライブ２のフロッピーディスクへ「MONDAI.16」という名前で保存して下さい。

<実行結果例>

```
save "2:MONDAI.16"
Ok
```

【命令のまとめ】

PAINT命令の一般形式と書き方例を次に示します。

<一般形式>

PAINT （X座標，Y座標），領域色，境界色

・座標(X，Y)の位置を中心にして，指定された境界色で囲まれる図形の内側の部分を指定した領域色で塗りつぶします。

・領域色および境界色はパレット番号で指定します。

・座標の値や領域色や境界色は，それぞれ変数を使って指定することもできます。

<書き方例>

① PAINT (100, 50), 4, 7

これは，座標(100, 50)の位置を中心にその周囲にある白色の線または領域の内側を緑色で塗ります。

② PAINT (200, 100), 2, 4

これは，座標(200, 100)の位置を中心にその周囲にある緑色の線または領域の内側を赤色で塗ります。

2.2.5 色の変更方法

問題 17 問題16で表示した円グラフの色を次のように変更して下さい。

青色 ──→ 赤色
赤色 ──→ 紫色
紫色 ──→ 緑色
緑色 ──→ 水色
水色 ──→ 黄色
黄色 ──→ 白色

【説明】

グラフィック画面に表示する図形の色は，パレット番号で指定します。「パレットというのは絵を描くときのパレットと同じで色を調合する道具である」ということは前に述べました。N_{88}-BASICで表示できる色は8色(黒，青，赤，紫，緑，水色，黄，白)です。この8色の各色にはカラーコードが付けられていますが，グラフィック画面へ表示する図形に付ける色はこのカラーコードを直接指定するのではなく，いったん，パレットへ色を定義しておいてそのパレットの色で指定するようにします。

	0	1	2	3	4	5	6	7
カラーコード	黒色	青色	赤色	紫色	緑色	水色	黄色	白色
	↓	↓	↓	↓	↓	↓	↓	↓
パレット番号	黒色	青色	赤色	紫色	緑色	水色	黄色	白色

⇩
グラフィック画面へ

パレットへの色の定義は，「COLOR」という命令で行います。マシンを立ち上げた直後はパレット番号の色とカラーコードは一致しています。パレットの中の色を変更すると，現在表示されている図形の色も変わります。

設問 17-1 パレット番号1の色(青色)を赤色に変更するため次の命令をキーインして実行して下さい。

＜実行する命令＞

```
COLOR=(1,2)
```

〔実行後の説明〕

① 命令実行すると，今まで青色で表示されていた図形が赤色に変わります。

② 赤色で表示されていた部分は変わりません。

③ この命令を実行した後,パレット番号1を指定して図形描いても表示される色は赤色になります。

設問 17-2 次の手順に従って，問題に示すように円グラフの色を変更して下さい。

〔手順〕

① 赤色を紫色に変えるために次の命令を実行します。

COLOR＝(2, 3)

② 紫色を緑色に変えるために次の命令を実行します。

COLOR＝(3, 4)

③ 緑色を水色に変えるために次の命令を実行します。

COLOR＝(4, 5)

④ 水色を黄色に変えるために次の命令を実行します。

COLOR＝(5, 6)

⑤ 黄色を白色に変えるために次の命令を実行します。

COLOR＝(6, 7)

〔実行後の説明〕

① 設問17-1でパレット番号1(青)は赤色に変更されています。したがって，この状態ではパレット番号1を指定しても2を指定しても赤色が表示されます。手順①に従って赤色を紫色に変更すると，パレット番号2で指定されている赤色が紫色に変わります。しかし，パレット番号1で指定されている赤色は変わりません。

設問 17-3 パレット番号が表す色を初期状態(パレット番号の色がカラーコードの色に一致している状態)に戻すために次の命令をキーインして実行して下さい。

<実行する命令>

```
FOR J=0 TO 7:COLOR=(J,J):NEXT J
```

〔実行後の説明〕

① 実行後のパレット番号0～7が表す色はカラーコードと同じ色になります。

② 通常グラフィック画面を利用するプログラムを作るときは，初めに必ずこの命令を書いて，パレット番号が表す色をカラーコードと同じ色にしておきます。

【命令のまとめ】

パレット番号の色を変更するCOLOR命令の一般形式と書き方例を次に示します。

<一般形式>

COLOR＝(パレット番号，カラーコード)

・パレット番号で指定したパレットにどの色を入れるかをカラーコードで指定します。

・一度指定したパレット番号の色は変更しない限り同じ色を表します。

<書き方例>

① COLOR＝(1，4)

これは，パレット番号1の表す色を緑色に変更します。

② COLOR＝(2，4)

これは，パレット番号2の表す色を緑色に変更します。

③ COLOR＝(4，2)

これは，パレット番号4の表す色を赤色に変更します。

問題　18　設問に従って画面全部を青色に変えて下さい。

【説　明】

テキスト画面に文字や数値に色を付けて表示するとき「COLOR」命令を使いました。この命令は前項のパレットの色を変更する「COLOR」命令と違ってテキスト画面の文字や数値に色を付ける役割のほか，グラフィック画面についても「フォアグラウンドカラー」や「バックグラウンドカラー」と呼ばれる画面の初期状態の色を変更する働きを持っています。

フォアグラウンドカラーというのはグラフィック命令を使って図形を表す場合で，パレット番号を省略したときに表示される色のことをいいます。

バックグラウンドカラーというのはグラフィック画面を「CLS」命令でクリアしたときに表示される色(背景の色)のことをいいます。また,「PRESET」命令でパレット番号を省略したときにもこの色が使われます。

設問 18-1　次の命令を実行して，バックグラウンドカラーを青色に変更して下さい。

<実行する命令>

```
COLOR ,1
```

〔実行後の説明〕

① グラフィック画面のバックグラウンドカラーは，青色に設定されましたが，画面上の色は変更されません。

② バックグラウンドカラーは，CLS命令でグラフィック画面をクリアするときに表示されます。

設問 18-2　グラフィック画面をクリアして下さい。

<実行する命令>

```
CLS 2
```

〔実行後の説明〕

① CLS命令の実行によってグラフィック画面が，設定されたバックグラウンドカラー(青色)で塗りつぶされていきます。

② 一度設定したバックグラウンドカラーは，再設定するまで有効です。

設問 18-3 次の命令を実行して，フォアグラウンドカラーを黄色に変更して下さい。

<実行する命令>

```
COLOR ,,,6
```

〔実行後の説明〕

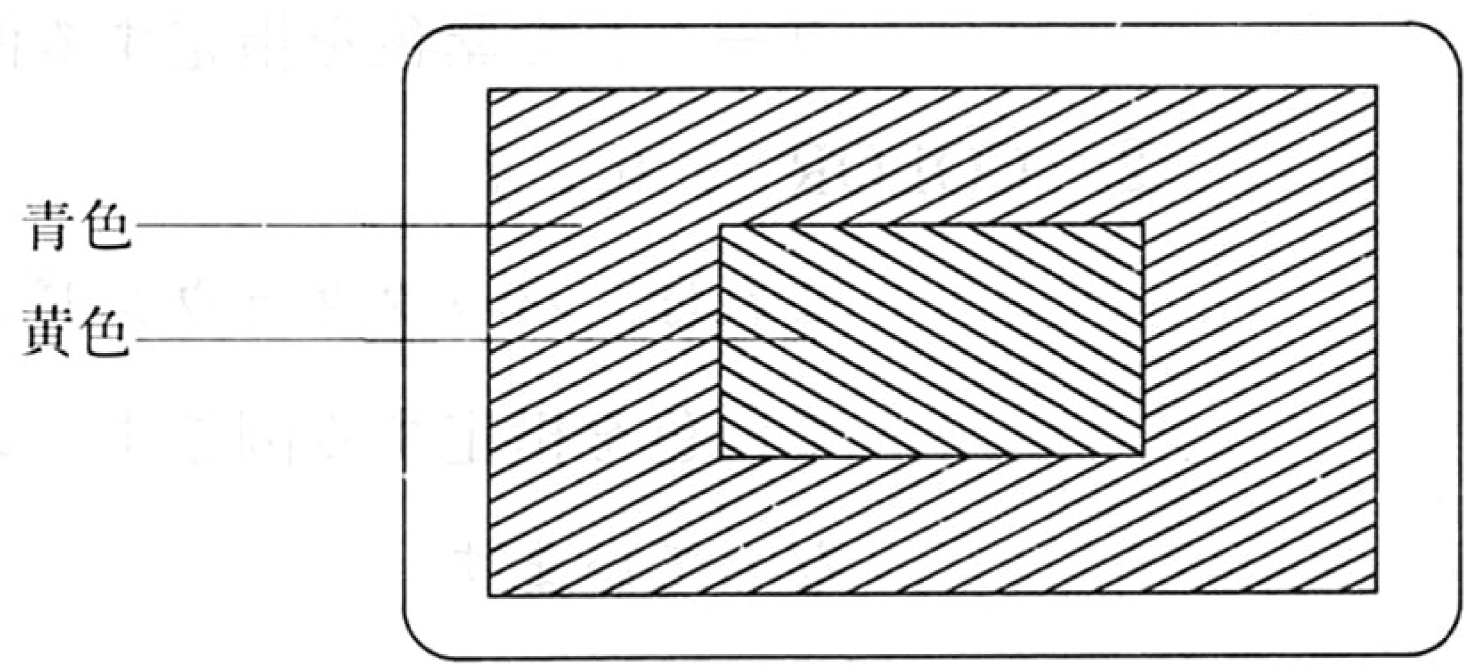
① フォアグラウンドカラーが黄色に変更されました。以降，パレット番号が省略されたときは，すべて黄色で表示されます。

設問 18-4 パレット番号を指定しないで，座標(200，50)と座標(500，150) を対角とする四角形を「BF」指定で描いて下さい。

〔操作説明〕

・命令は，「LINE (200，50)-(500，150)，，BF」をキーインします。

<実行結果例>



〔実行後の説明〕

① パレット番号を省略してグラフィック画面に図形を描くと，現在設定されているフォアグラウンドカラーの色(ここでは黄色)で表示されます。

設問 18-5 フォアグラウンドカラーを白，バックグラウンドカラーを黒の初期状態に戻した後，画面をクリアして下さい。

<実行結果例>

```
COLOR ,0,,7
Ok
CLS 3
          ←―――――― ここで画面がクリアされる
Ok
```

【命令のまとめ】

フォアグラウンドカラーやバックグラウンドカラーを変更するCOLOR命令の一般形式と書き方例を次に示します。

<一般形式>

COLOR ファンクションコード，バックグラウンドカラー，，フォアグラウンドカラー

・ファンクションコードは，テキスト画面についての色の指定等を行います。

・バックグラウンドカラーは，グラフィック画面の背景の色を何色にするかを決めます。省略したときは，既に指定された色になります。(通常は黒色です)

・フォアグラフィックは，グラフィック命令のパレット番号を省略したときに表示する色を指定します。(通常は白色です)

・N_{88}-BASICでは，バックグラウンドカラーとフォアグラウンドカラーの間にボーダーカラー(画面の枠の色)の指定を行うことができます。しかし，PC-8801mkIIではボーダーカラーを使うことはできません。この指定は，PC-8801との互換を保つためのもので，PC-8801mkIIでは意味を持ちません。したがって，省略します。

<書き方例>

① COLOR ，1,，3

これは，バックグラウンドカラーとして青色を指定し，フォアグラウンドカラーとして紫色を指定する例です。

② COLOR ，0,，7

これは，バックグラウンドカラーとして黒色，フォアグラウンドカラーとして白色を指定する例です。システムを起動したとき(初期状態)はこの状態になっています。

2.3 画面モードの設定方法

グラフィック画面には，カラーモードや白黒モードのような，使い方の異なるモードがあることについては既に説明しました。ここでは，このモードを変更する命令について学習します。

問題 19 白黒モードで3枚の各プレーンに次のような円を描いた後,画面をカラーモードにして下さい。

プレーン1	中心(200，50)，半径100
プレーン2	中心(300，50)，半径100
プレーン3	中心(250，100)，半径100

円の中も塗りつぶすようにして下さい。

【説　明】

グラフィック画面のモードには「白黒モード」，「カラーモード」，「高分解能白黒モード」の三つがあります。これらのモードの切り替えには「SCREEN」という命令を使います。

白黒モードでは，3枚のプレーン(ページ)を別々に使用することができます。ここでは，問題にあるような円を各ページごとに描いていきます。白黒モードでは，図形を描くページのことを「アクティブページ」，画面に表示するページのことを「ディスプレイページ」と呼びます。アクティブページとディスプレイページは，同時に違うものを指定することができます。ですから，ページ1に図形を描き込みながら，ページ2を表示しておくといったことができます。

カラーモードでは,3枚のページすべてがアクティブページおよびディスプレイページになります。したがって，各ページを別々に使用することはできません。

モードの切り替えを行っても，前のモードで描き込んだ図形は消えません。したがって，白黒モードで各プレーンに円を描き込んだ後，カラーモードに切り替えると描き込んだページに従ってカラーで表示されます。

設問 19-1 画面をクリアした後，次の命令を実行して画面を白黒モードにし，アクティブページとディスプレイページを1に設定して下さい。

<実行する命令>

```
SCREEN 1,,0,1
```

〔実行後の説明〕

① この命令の実行によって，画面モードは「1」(白黒モード)，アクティブページは「0」(ページ1)，ディスプレイページは「1」(ページ1)が設定されます。

設問 19-2 座標(200，50)を中心とする半径100の円を描いた後，その円を塗りつぶしして下さい。

〔操作説明〕

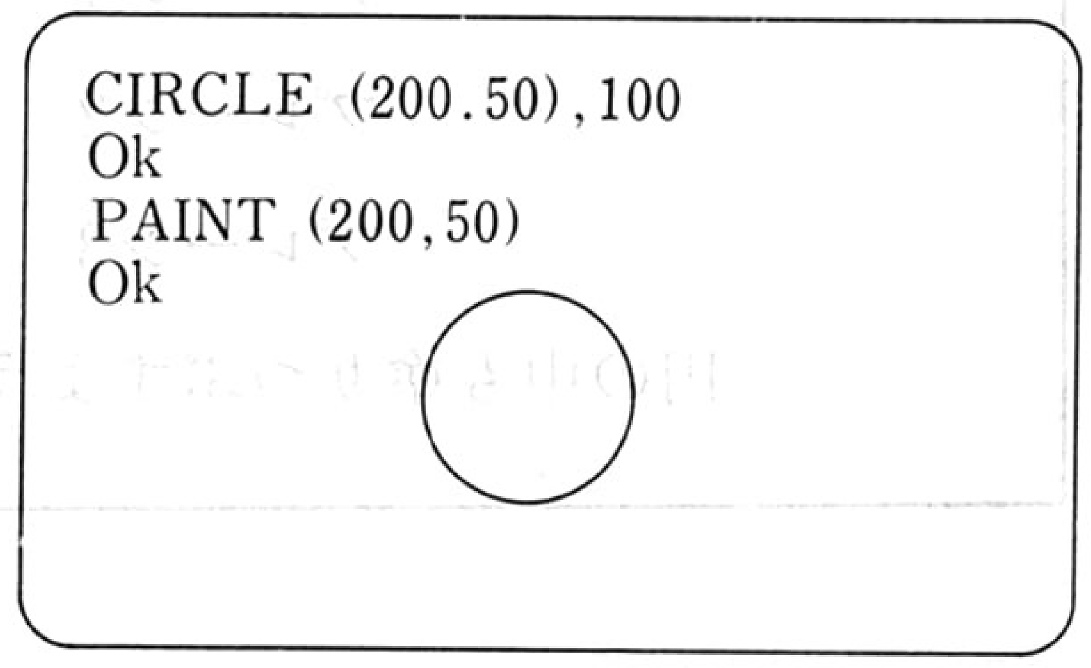
・命令は

「CIRCLE (200，50)，100」

「PAINT (200，50)」

をキーインします。

<実行結果例>



〔実行後の説明〕

① ページ1に白く塗りつぶされた半径100の円を描きます。

② 画面上にはグラフィック画面のページ1だけが表示されています。

設問 19-3 次の命令を実行して，アクティブページとディスプレイページをページ2に設定して下さい。

<実行する命令>

```
SCREEN ,,1,2
```

〔実行後の説明〕

① 画面モードは変更しないので省略します。

② アクティブページは「1」(ページ2)，ディスプレイページは「2」(ページ2)が設定されます。

③ 画面上に表示されるグラフィック画面はページ2になります。したがって，設問19-2でページ1に描いた円は，画面には表示されません。

設問 19-4　座標(300，50)を中心とする半径100の円を描いた後，その円を塗りつぶして下さい。

〔操作説明〕

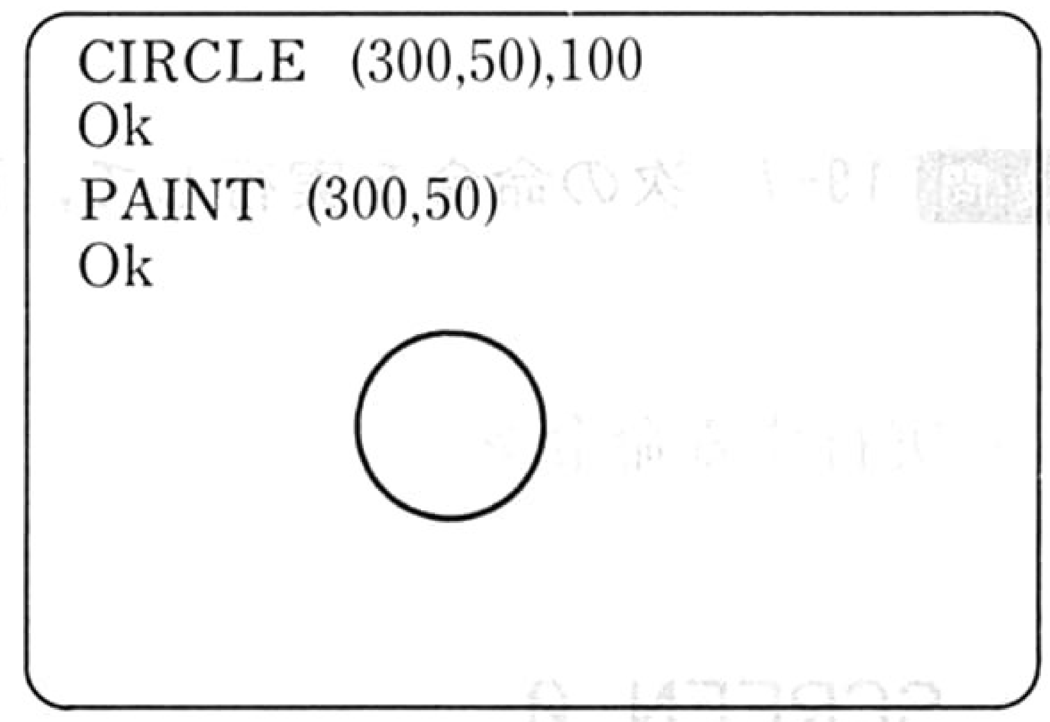
・命令は

「CIRCLE　(300，50)，100」

「PAINT　(300，50)」

をキーインします。

＜実行結果例＞



〔実行後の説明〕

① ページ2に白く塗りつぶされた半径100の円を描きます。

② 画面上にはグラフィック画面のページ2だけが表示されます。

設問 19-5　次の命令を実行して，アクティブページとディスプレイページをページ3に設定して下さい。

＜実行する命令＞

```
SCREEN ,,2,4
```

〔実行後の説明〕

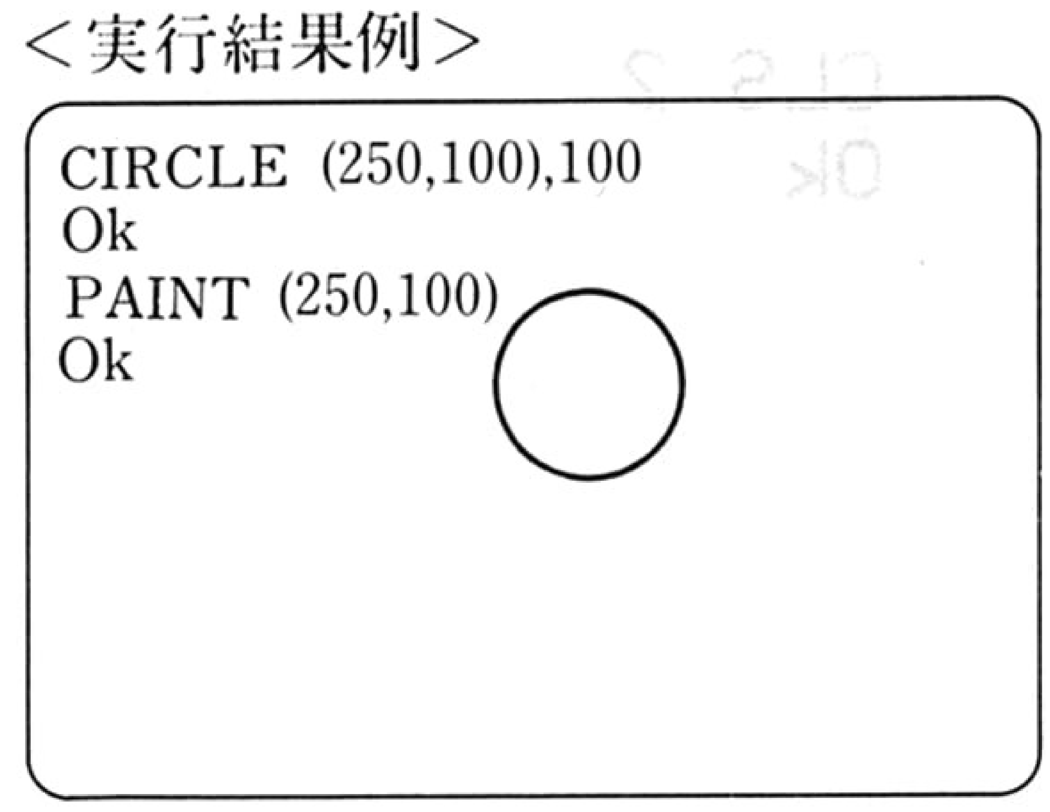
① 画面モードは変更しないので省略します。

② アクティブページは「2」(ページ3)，ディスプレイページは「4」(ページ3)が設定されます。

③ 画面上に表示されるグラフィック画面はページ3になります。したがって，設問19-4と19-2でページ2とページ1に描いた円は，画面には表示されません。

設問 19-6　座標(250，100)を中心とする半径100の円を描いた後，その円を塗りつぶして下さい。

＜実行結果例＞



〔操作説明〕

・命令は

「CIRCLE　(250，100)，100」

「PAINT　(250，100)」

をキーインします。

〔実行後の説明〕

① ページ3に白く塗りつぶされた半径100の円を描きます。

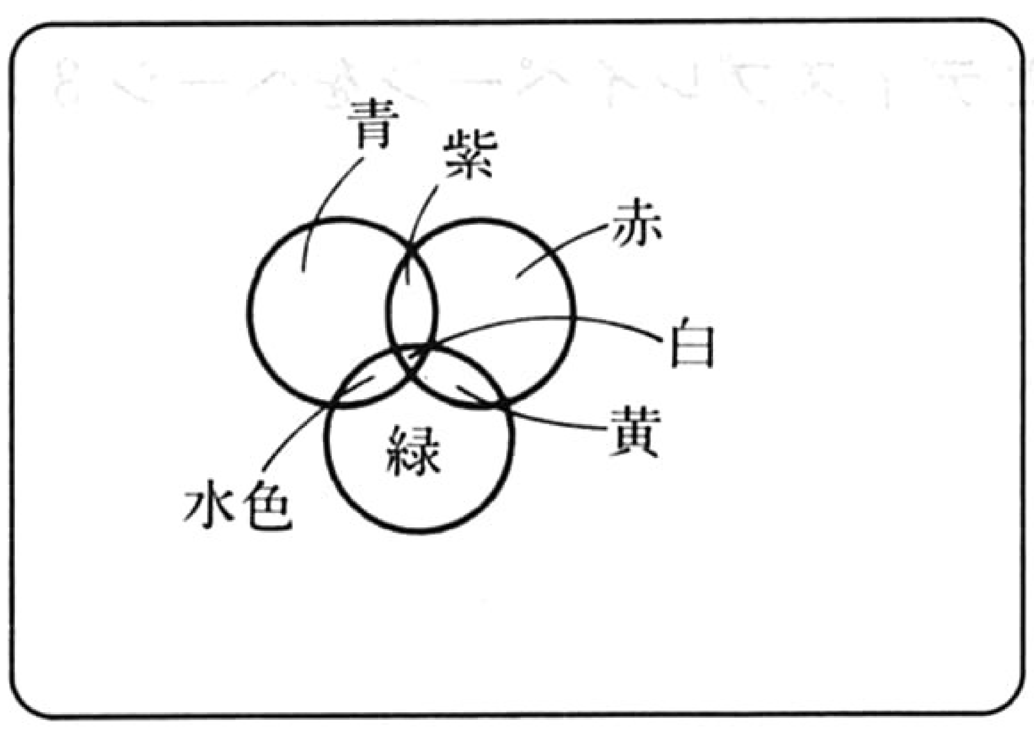
② 画面上にはグラフィック画面のページ3だけが表示されます。

設問 19-7 次の命令を実行して，画面をカラーモードにして下さい。

<実行する命令>

```
SCREEN 0
```

<実行結果例>



〔実行後の説明〕

① 画面のモードをカラーモードに変更しました。

② カラーモードにすることによって白黒モード時に各ページに描き込んだ円が，重なって表示されます。

③ 表示された円の色は，光の三原色です。円自身の色や重なっている部分の色と円を描き込んだページとの関連を考えて下さい。

設問 19-8 グラフィック画面をクリアして下さい。

<実行結果例>

```
CLS 2
Ok
```

【命令のまとめ】

グラフィック画面のモードを変更したり，アクティブページやディスプレイページを変更したりするSCREEN命令の一般形式と書き方例を次に示します。

＜一般形式＞

SCREEN　画面モード，画面スイッチ，アクティブページ，ディスプレイページ

・画面モードは，グラフィック画面を白黒モードで使うか等の条件を次の値で指定します。省略すると，既に指定されていたモードが維持されます。

0　カラーモード　(640×200ドット)

1　白黒モード　(640×200ドット×3ページ)

2　高分解能白黒モード　(640×400ドット)

・画面スイッチはグラフィック画面に既に表示されている図形を一時的に消したり再び表示したりするための「グラフィックマスクスイッチ」と，図形をグラフィックRAM(図形を記憶しておくメモリー)へ書き込む速度を高速にするか否かをきめる「高速書き込みスイッチ」の状態を次の値で指定します。省略すると，既に指定されていたスイッチの状態が維持されます。

0　両スイッチともOFFにするとき(通常の速度で表示します)

1　高速書き込みスイッチだけONにするとき(高速で表示します)

2　グラフィックマスクスイッチだけONにするとき
(通常の速度でグラフィックRAMへ書くだけで表示は行いません)

3　両方ともONにするとき
(高速でグラフィックRAMへ書くだけで表示は行いません)

※　高速書き込みスイッチをONにすると，図形を表示するとき画面がチラチラするため，よほど高速性が必要なとき以外は使わないようにします。したがって，通常は「0」か「2」のどちらかを使います。

・アクティブページは，白黒モードのときだけ意味を持つもので，図形をどのページに書き込むかを次の値で指定します。省略すると，既に指定されていたページに書き込むことになります。

0　ページ1へ書き込むとき

1　ページ2へ書き込むとき

2　ページ3へ書き込むとき

※　カラーモードや高分解能白黒モードのときに指定しても意味を持ちません。

・ディスプレイページは，アクティブページの指定と同様に白黒モードのときだけ意味を持つもので，どのページを画面に表示するかを次の値で指定します。

0　どのページの図形も表示しないとき

1　ページ1の図形だけ表示するとき
2　ページ2の図形だけ表示するとき
3　ページ1とページ2の図形を合成して表示するとき
4　ページ3の図形だけ表示するとき
5　ページ1とページ3の図形を合成して表示するとき
6　ページ2とページ3の図形を合成して表示するとき
7　ページ1とページ2とページ3の全部の図形を合成して表示するとき

※　カラーモードや高分解能白黒モードのときに指定しても意味を持ちません。

<書き方例>

①　SCREEN　0，0

これは，画面モードをカラーモードに変更して，画面スイッチをOFFにします。

②　SCREEN　0，2

これは，画面モードをカラーモードに変更して，グラフィックマスクスイッチだけONにします。したがって，グラフィック画面には図形を表示します。

③　SCREEN　1，0，2，4

これは，画面モードを白黒モードに変更して，更に画面スイッチをOFFにし，ページ3のグラフィックRAMへの書き込みとグラフィック画面への表示をします。

④　SCREEN　1，，2，4

これは，画面スイッチを省略，つまり既に指定されていた状態にしたままで他は③と同じ状態にします。

⑤　SCREEN　2，0

これは，画面モードを高分解能白黒モードに変更して，画面スイッチをOFFにします。

2.4　絶対座標と相対座標

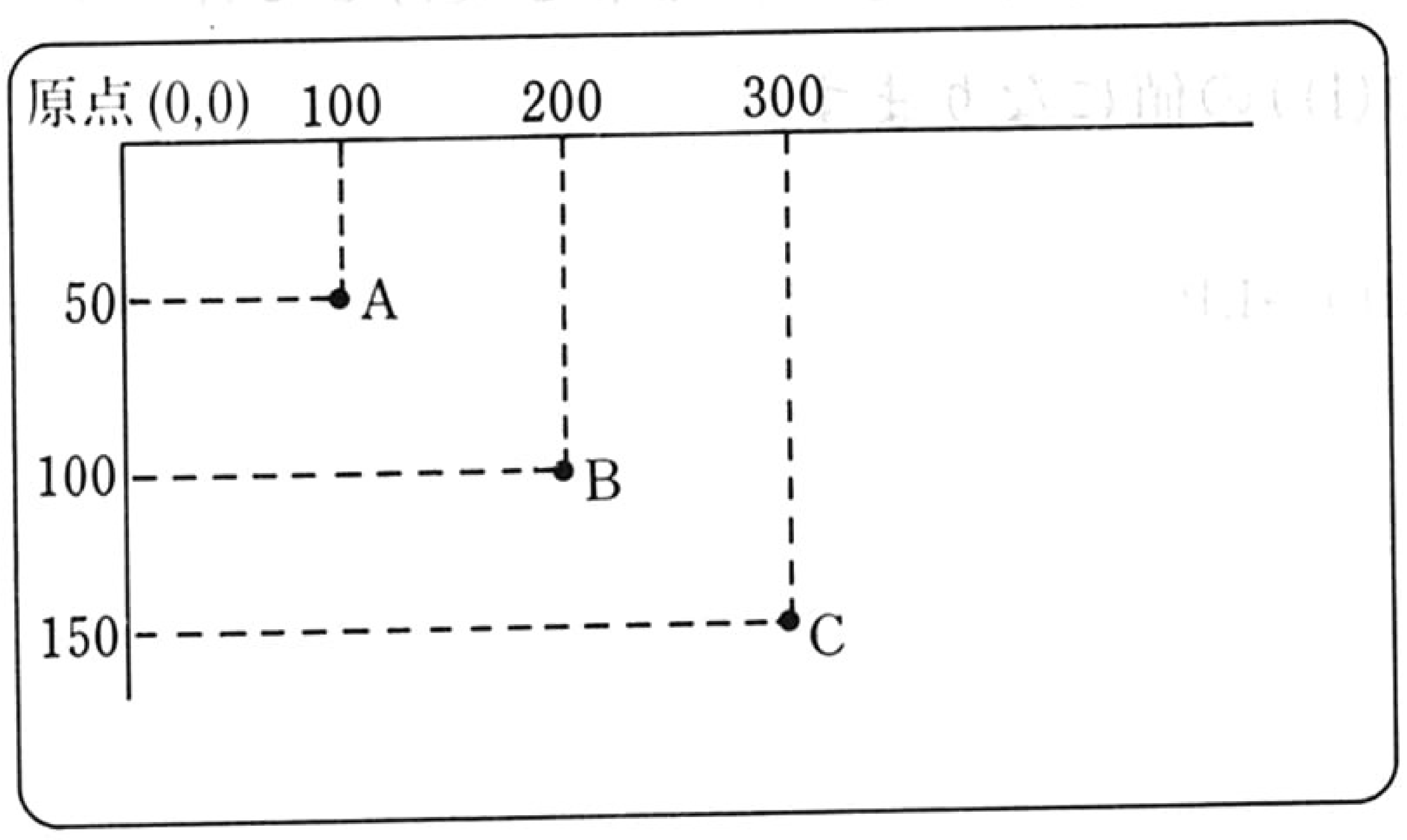
今までの学習でグラフィック画面に図形を描くときに指定した座標は，常に画面上の絶対的な位置を示すもので,「絶対座標」と呼ばれます。N_{88}-BASICには，このほかに相対的な位置が指定できる相対座標があります。

「相対座標」とは，画面内のある1点を基点として，X方向(横方向)，Y方向(縦方向)にどれだけの位置にあるかによって決められる座標です。相対座標の原点となる基点のことを,「LP(Last referenced Point)」と呼びます。

「LP」は最後に参照された位置という意味で，直前に行われたグラフィック命令での座標になります。

相対座標の指定は，通常の絶対座標の指定と区別するために，グラフィック命令での座標指定の前に必ず「STEP」を付けます。

たとえば，次の図のように座標A(100，50)，B(200，100)，C(300，150)ような等間隔の位置に点を表示する場合，絶対座標による指定では一つずつ座標の指定が変わりますが，相対座標で指定する座標は一つでも表すことができます。



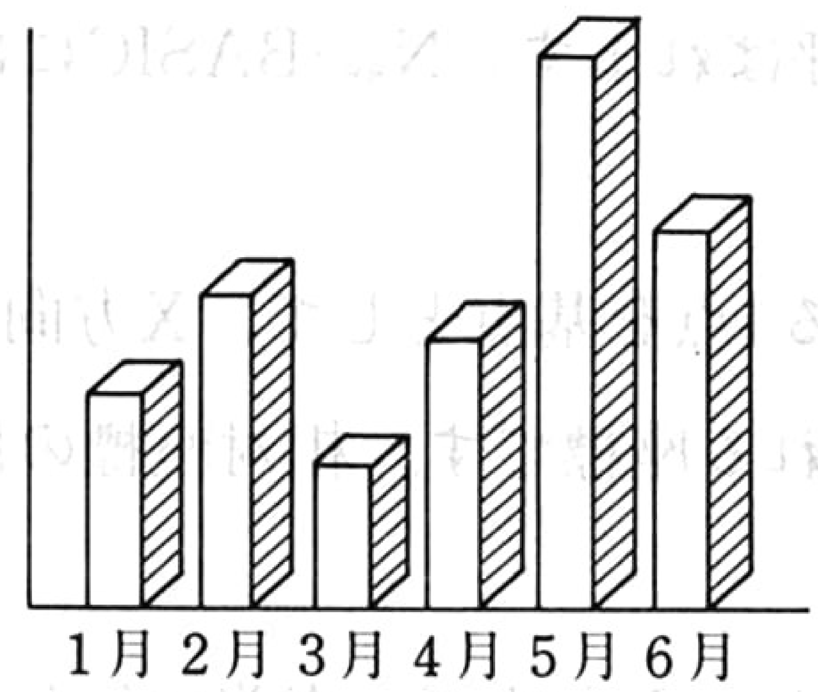
<絶対座標指定による表示>

```
PSET (100, 50)
PSET (200, 100)
PSET (300, 150)
```

<相対座標指定による表示>

```
FOR I=1 TO 3
    PSET STEP(100, 50)
NEXT I
```

問題　20　問題12で作成した棒グラフを表示するプログラムを改造して，次のように棒グラフに陰影を付け立体棒グラフを表示して下さい。



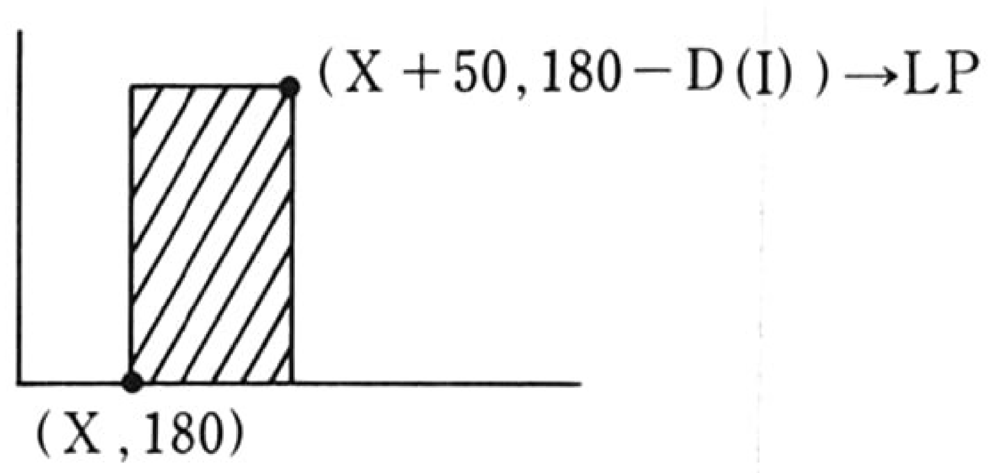
【説　明】

平面的な棒グラフに立体感をつけるため，相対座標を使って陰影を付けます。

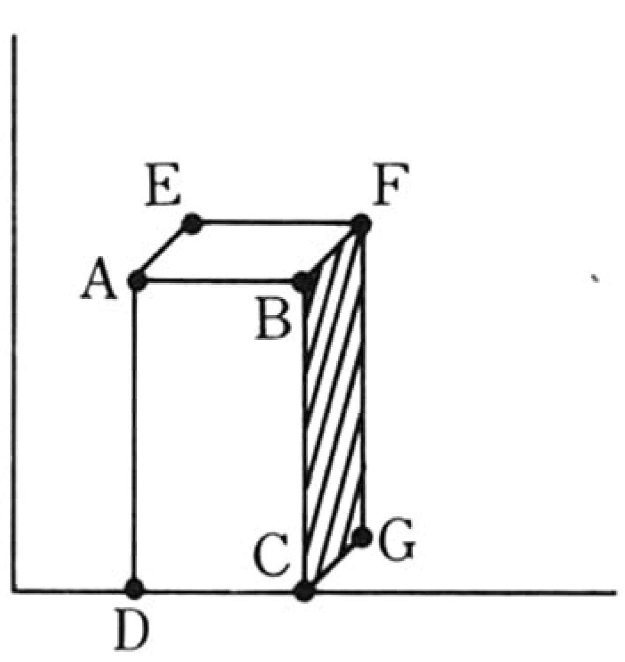
問題12のプログラムで，棒グラフを表示する命令は次のようになっています。

```
LINE　(X, 180)-(X+50, 180-D(I)), I+1, BF
```

この命令を実行して表示される棒グラフは次のようなものです。命令を実行して棒グラフを表示した後の基点となるLPは(X+50, 180−D(I))の値になります。



立体感を出すために陰影を付ける線を次のようにひきます。



・点A，B，Cの各点からX軸方向に＋20，Y軸方向に−15の位置に点E，F，Gをとります。

・平面棒グラフを表示した後，LPは点Bにあります。したがって，点B→点F→点E→点Aと線を引いた後，点C→ 点G→点Fへと線を引きます。

・点Cの座標は，(X+50, 180)になります。

【 プログラムの修正 】

設問 20-1 ドライブ2のフロッピーディスクから「MONDAI.12」というプログラムをメモリーにロードして下さい。

＜実行結果例＞

```
load "2:MONDAI.12"
Ok
```

設問 20-2 行番号240を表示して，次のように修正して下さい。

＜修正リスト＞

```
240   LINE (X,180)-(X+50,180-D(I)),I+1,B
```

〔修正後の説明〕

① 表示する立体棒グラフは，側面に色を塗って陰影を付けるようにします。したがって，前面の四角形は塗りつぶさないようにします。

設問 20-3 次の命令を追加して下さい。

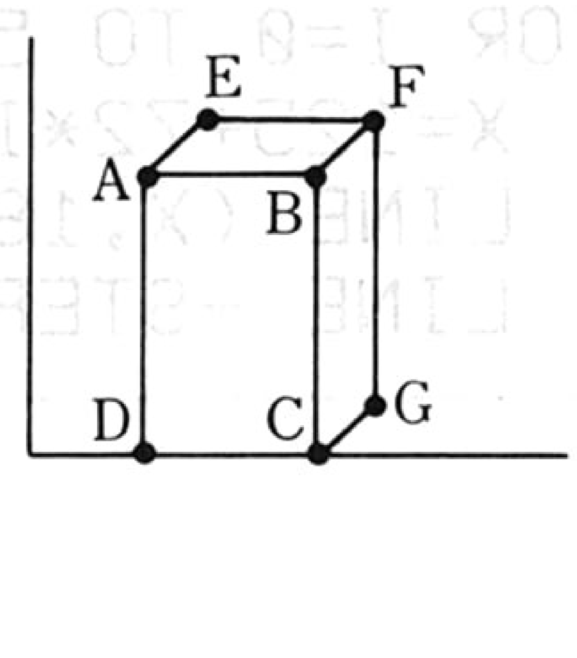
＜追加する命令＞

```
241   LINE -STEP(20,-15),I+1
242   LINE -STEP(-50,0),I+1
243   LINE -(X,180-D(I)),I+1
244   LINE (X+50,180)-STEP(20,-15),I+1
245   LINE -STEP(0,-D(I)),I+1
246   PAINT STEP(-1,16),I+1,I+1
```

〔追加後の説明〕

① 行番号241は，行番号240で四角形を表示した後のLPの位置からX軸方向に＋20，Y軸方向に－15の位置（点Bから点F）に線を引きます。

この命令を実行した後，LPは点Fに移動します。



② 行番号242は，現在のLP(点F)から点Eに直線を引きます。

この命令を実行した後，LPは点Eに移動します。

③ 行番号243は，現在のLP(点E)から点Aに直線を引きます。

点Aの位置は，相対座標を使わなくてもわかります。もちろん相対座標を使って，「LINE −STEP(−20, +15), I+1」と指定しても構いません。

④ 行番号244は，点Cから点Gに直線を引きます。この命令を実行するとき，LPは点Aの位置にあります。そのためにここでは点Cを絶対座標で指定してLPを点Cに移動した後，そこから点Gの相対座標を指定して線を引いています。LPの移動は，グラフィック命令によって行われますが，「POINT」という命令でも移動させることができます。POINT命令を使って，LPを点Cに移動させて点Cと点Gを結ぶ直線を引く例を次に示します。

POINT (X+50, 180)

LINE −STEP(20, +15), I+1

このように書いても，行番号244と同じような結果が得られ，LPは点Gに移動します。

⑤ 行番号245は，点Gから点Fへ直線を引きます。

その結果，LPは点Fに移動します。

⑥ 行番号246は，四角形BCGFを塗りつぶすためのPAINT命令です。

【 プログラムの確認 】

設問 20-4 プリンタのリストをプリンタに印字して，正しくキーインできているかどうか確認して下さい。

<確認用リスト>

```
100 DIM D(5)
110 WIDTH 80,25:CONSOLE ,,0:SCREEN 0,0
120 CLS 3
130 FOR I=0 TO 5
140   PRINT USING " #月 ";I+1;
150   INPUT D(I)
160 NEXT I
170 LINE (100,180)-(100,0)
180 LINE (100,180)-(570,180)
190 FOR Y=180 TO 0 STEP -18
200   LINE (99,Y)-(101,Y)
210 NEXT Y
220 FOR I=0 TO 5
230   X=125+72*I
240   LINE (X,180)-(X+50,180-D(I)),I+1,B
241   LINE -STEP(20,-15),I+1
```

(つづく)

```
242    LINE -STEP(-50,0),I+1
243    LINE -(X,180-D(I)),I+1
244    LINE (X+50,180)-STEP(20,-15),I+1
245    LINE -STEP(0,-D(I)),I+1
246    PAINT STEP(-1,16),I+1,I+1
250    X=17+9*I
260    LOCATE X,23:PRINT USING "##月";I+1
270 NEXT I
280 INPUT "ツヅケマスカ (y/n)";YN$
290 IF YN$="y" OR YN$="Y" THEN 120
300 CLS 3:END
```

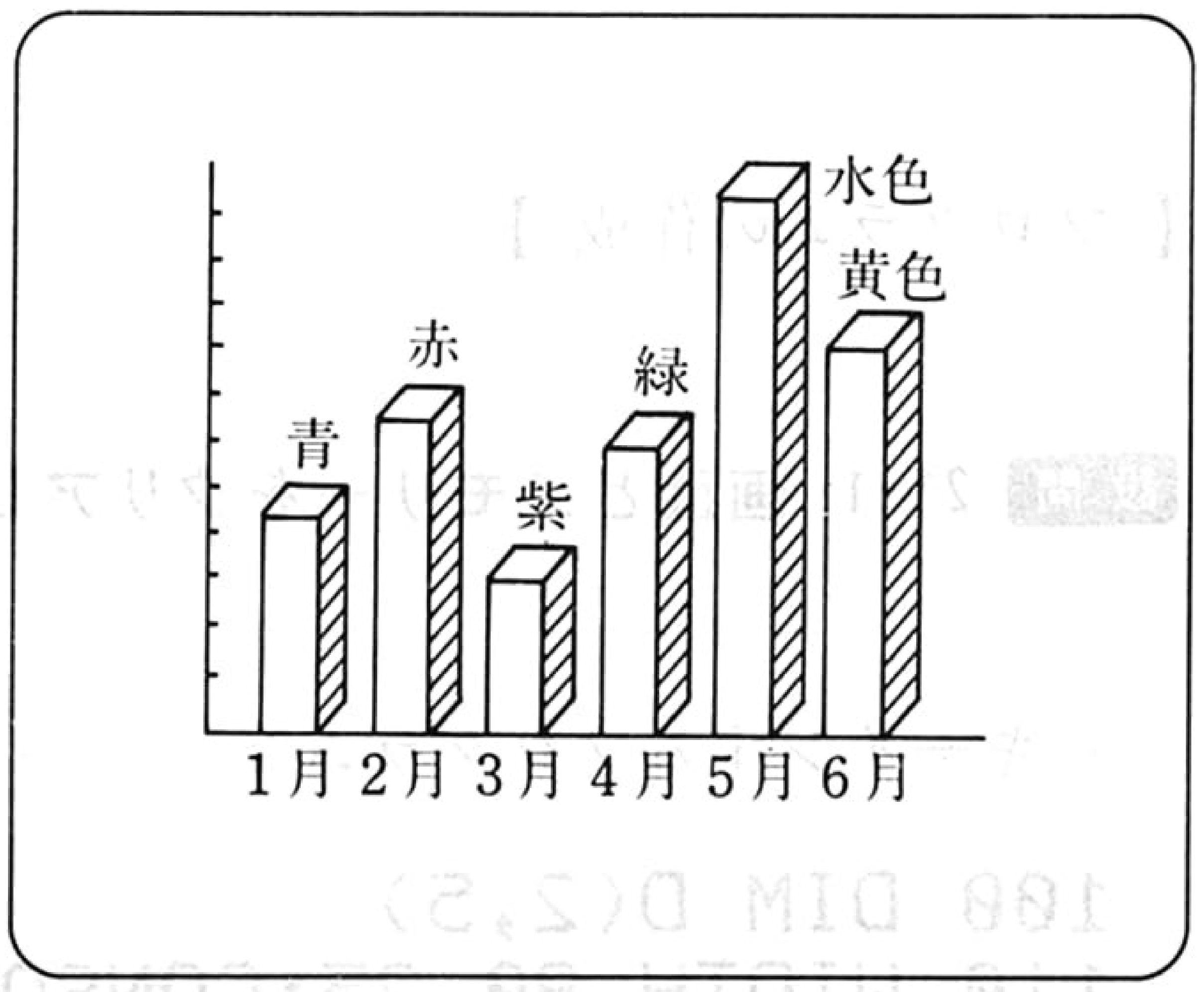
設 問 20-5　正しくキーインできていたら，プログラムを実行して下さい。

〔操作説明〕

入力する売上データは次のとおりです。

1月	70
2月	100
3月	50
4月	90
5月	180
6月	120

＜実行結果例＞



〔実行後の説明〕

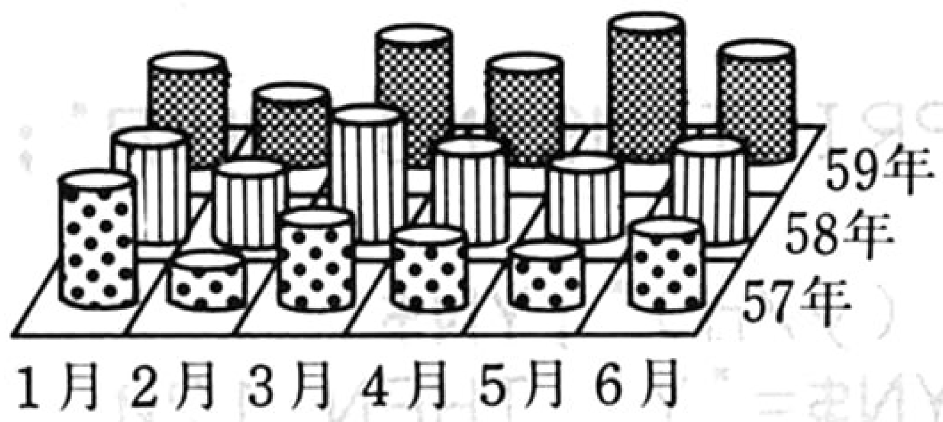
① このプログラムは，入力した値をそのまま棒グラフの長さに使っています。したがって，180以上の値を入力すると，グラフが画面からはみ出してしまうとともに，PAINT命令の実行とともに「Illegal　functin　call」エラーが発生します。

設 問 20-6　このプログラムをドライブ2のフロッピーディスクへ「MONDAI.20」という名前で保存して下さい。

＜実行結果例＞

```
save "2:MONDAI.20"
Ok
```

問題　21　問題20を応用して半期分の売上データの割合を過去3年間の同期のものと比較できるように，過去3年間のデータを入力して，次のような三次元の円柱グラフを表示しなさい。



【 説　明 】

三次元グラフの作成は，二次元グラフと同様座標軸を描いた後，奥の方のグラフ表示から行います。

ここでは，円柱を使ってグラフ化しています。

【 プログラムの作成 】

設問 21-1 画面とメモリーをクリアしてから，次のプログラムをキーインして下さい。

<キーインするプログラム>

```
100 DIM D(2,5)
110 WIDTH 80,25:CONSOLE ,,0:SCREEN 0,0
120 CLS 3
130 FOR I=0 TO 2
140   PRINT USING " ##年";I+57
150   FOR J=0 TO 5
160     PRINT USING "      #月";J+1;
170     INPUT D(I,J)
180     IF D(I,J)>MAX THEN MAX=D(I,J)
190 NEXT J,I
200 FOR X=0 TO 3
210   LINE (100+X*18,180-X*24)-STEP(432,0)
220 NEXT X
230 FOR Z=0 TO 6
240   LINE (100+72*Z,180)-(154+72*Z,108)
250 NEXT Z
260 FOR I=0 TO 2
270   FOR J=0 TO 5
280     X=(181-18*I)+72*J:Y=120+24*I
290     H=INT((D(I,J)*100)/MAX)
300     C=6
310     IF C=J+1 THEN C=1
```

(つづく)

```
320      CIRCLE (X,Y-H),25,C,,,.25
330      PAINT (X-24,Y-H),J+1,C
340      PAINT (X-24,Y-H),7,C
350      LINE (X-25,Y-H)-STEP(0,H),C
360      LINE (X+25,Y-H)-STEP(0,H),C
370      CIRCLE (X,Y),25,C,3.14159,6.28318,.25
380      PAINT (X-24,Y),J+1,C
390      CIRCLE (X,Y-H),25,7,,,.25
400      LINE (X-25,Y-H)-STEP(0,H),7
410      LINE (X+25,Y-H)-STEP(0,H),7
420      CIRCLE (X,Y),25,7,3.14159,6.28318,.25
430    NEXT J
440    LOCATE 74-I*2,15+I*3:PRINT USING "##年";I+57
450 NEXT I
460 FOR I=0 TO 5
470    LOCATE 16+I*9,23:PRINT USING "#月";I+1
480 NEXT I
490 LOCATE 0,23
```

〔キーイン後の説明〕

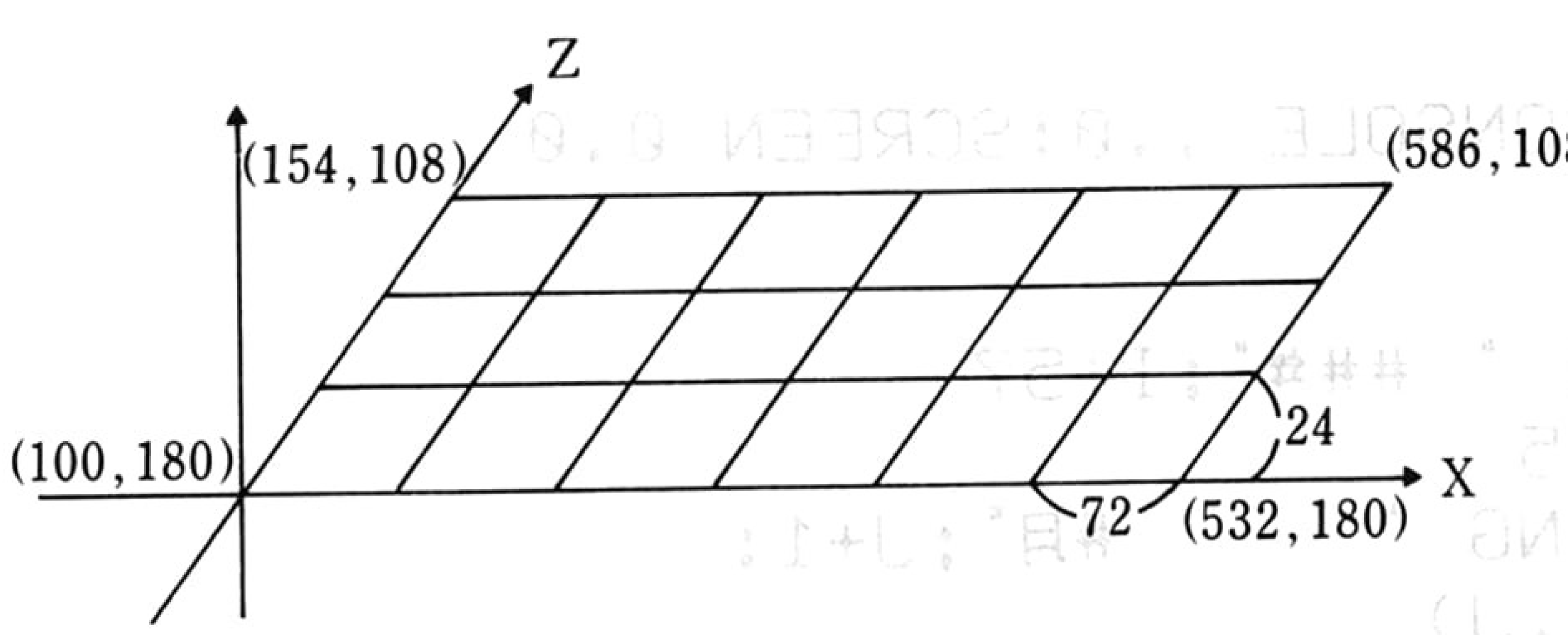
① 行番号100では，過去3年間の半期分の売上データを入れておく変数として，2次元の配列Dを確保しています。

② 行番号130～190は，売上データの入力を行っています。

③ 行番号200～220は，座標軸のX方向の線を描いています。

④ 行番号230～250は，座標軸のZ方向の線を描いています。

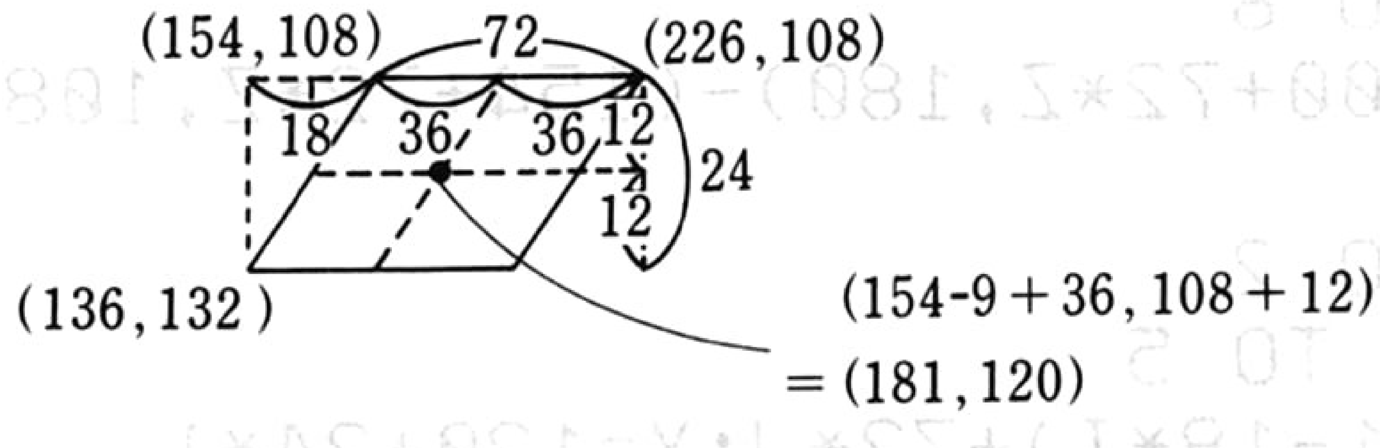
⑤ 行番号200～250で描かれる座標軸は次のようになります。



⑥ 行番号260～450は，円柱グラフを57年分から順に表示しています。

⑦ 行番号280は，座標軸の中に円柱を表示するときの中心の座標を求めています。

(例) 57年の1月分の円柱グラフの中心座標は次のようになります。



⑧　行番号290は，各データの最大値に対する割合を変数Hに求めています。求めた割合は，そのままグラフの高さに利用します。

⑨　行番号300と行番号310は，円柱グラフの輪郭の色を決めています。

⑩　行番号320は，円柱グラフの上部の楕円を描いています。

⑪　行番号330と行番号340は，楕円の中を白く塗りつぶすようにしています。

⑫　行番号350と行番号360は，円柱の両側の線を描いています。

⑬　行番号370は，円柱グラフの下部の円弧を行番号280で求めた座標を中心にして描いています。

⑭　行番号380は，円柱グラフに各月ごとに違った色を塗っています。

⑮　行番号390～420は，円柱グラフの輪郭を白色で描き直しています。最初から白色で描かないのは，座標軸が白色のためにPAINT命令を実行する場合の境界色の設定ができないからです。

⑯　行番号440は，座標軸の横に年を表示しています。

⑰　行番号460～480は，座標軸の下に月を表示しています。

【 プログラムの確認 】

設問 21-2　プログラムのリストをプリンタに印字して，正しくキーインできているかどうか確認して下さい。

<確認用リスト>

```
100 DIM D(2,5)
110 WIDTH 80,25:CONSOLE ,,0:SCREEN 0,0
120 CLS 3
130 FOR I=0 TO 2
140   PRINT USING " ##年";I+57
150   FOR J=0 TO 5
160     PRINT USING "      #月";J+1;
170     INPUT D(I,J)
180     IF D(I,J)>MAX THEN MAX=D(I,J)
190 NEXT J,I
200 FOR X=0 TO 3
210   LINE (100+X*18,180-X*24)-STEP(432,0)
220 NEXT X
230 FOR Z=0 TO 6
240   LINE (100+72*Z,180)-(154+72*Z,108)
250 NEXT Z
260 FOR I=0 TO 2
270   FOR J=0 TO 5
280     X=(181-18*I)+72*J:Y=120+24*I
```

（つづく）

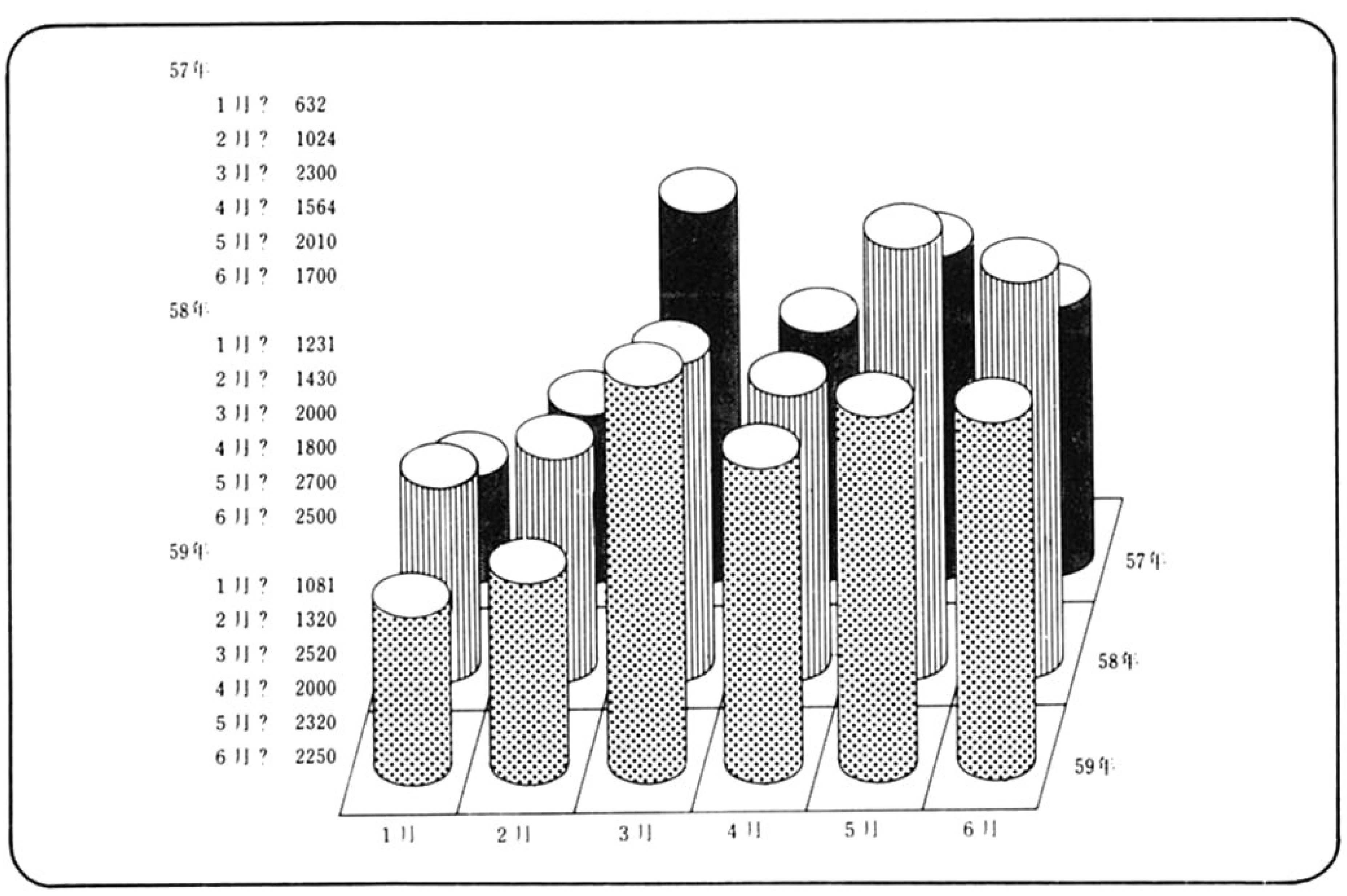
```
290       H=INT((D(I,J)*100)/MAX)
300       C=6
310       IF C=J+1 THEN C=1
320       CIRCLE (X,Y-H),25,C,,,.25
330       PAINT (X-24,Y-H),J+1,C
340       PAINT (X-24,Y-H),7,C
350       LINE (X-25,Y-H)-STEP(0,H),C
360       LINE (X+25,Y-H)-STEP(0,H),C
370       CIRCLE (X,Y),25,C,3.14159,6.28318,.25
380       PAINT (X-24,Y),J+1,C
390       CIRCLE (X,Y-H),25,7,,,.25
400       LINE (X-25,Y-H)-STEP(0,H),7
410       LINE (X+25,Y-H)-STEP(0,H),7
420       CIRCLE (X,Y),25,7,3.14159,6.28318,.25
430     NEXT J
440     LOCATE 74-I*2,15+I*3:PRINT USING "##年";I+57
450   NEXT I
460   FOR I=0 TO 5
470     LOCATE 16+I*9,23:PRINT USING "#月";I+1
480   NEXT I
490   LOCATE 0,23
```

設問 21-3　正しくキーインできていたら，次の売上データでプログラムを実行して下さい。

＜実行する売上データ＞

	1月	2月	3月	4月	5月	6月
57年	632	1,024	2,300	1,564	2,010	1,700
58年	1,231	1,430	2,000	1,800	2,700	2,500
59年	1,081	1,320	2,520	2,000	2,320	2,250

＜実行結果例＞



〔実行後の説明〕

① データとして「0」を入力しても，白い楕円は表示されます。

設問 21-4 このプログラムをドライブ2のフロッピーディスクへ「MONDAI.21」という名前で保存して下さい。

＜実行結果例＞

```
save "2:MONDAI.21"
Ok
```

【 命令のまとめ 】

LP(Last referenced Point)を変更するPOINT命令の一般形式と書き方例を次に示します。

＜一般形式＞

POINT (X, Y)

- 指定された座標位置に，相対座標で指定を行うときの基点となるLPを移動します。
- 指定する座標は，ワールド座標系になります。

＜書き方例＞

① POINT (300, 100)

これは，現在のLPを座標(300, 100)の位置に移動します。

② POINT STEP(10, 20)

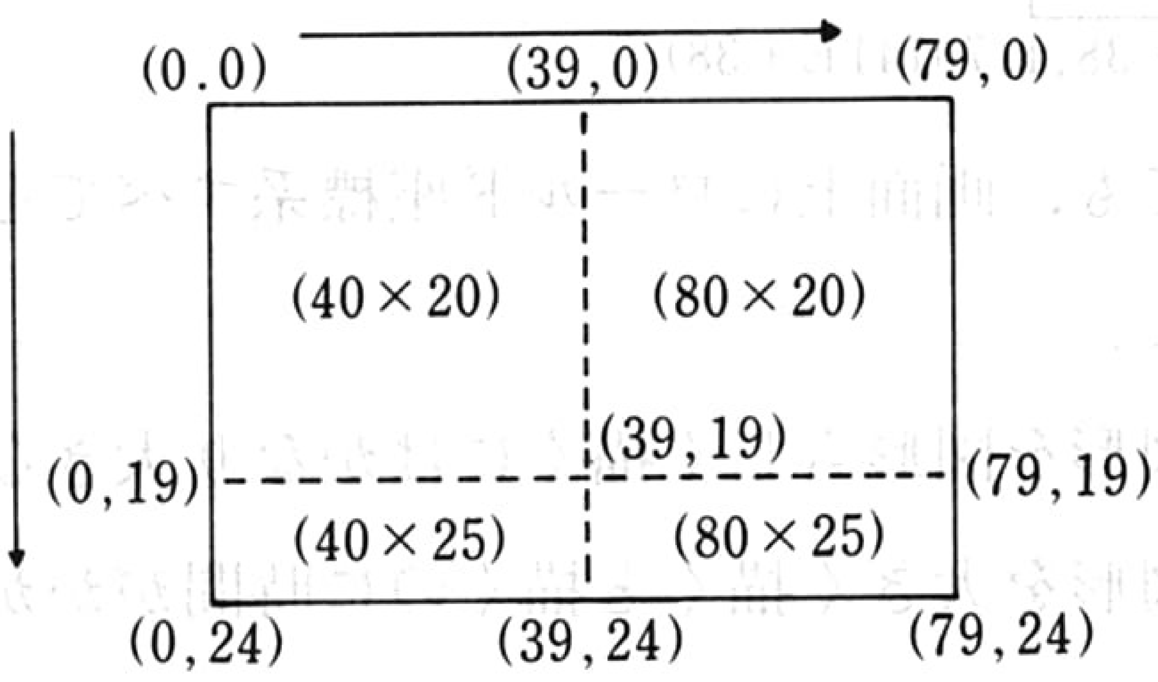
これは，現在のLP位置からX方向へ10，Y方向へ20の位置へLPを移動します。

2.5　座標系について

今までの学習で，グラフィック画面に図形を表示する場合座標を用いていました。グラフィック画面に展開されているこの座標のことを，「グラフィック座標系」と呼びます。(テキスト画面に展開される座標は「キャラクタ座標系」と呼びます)

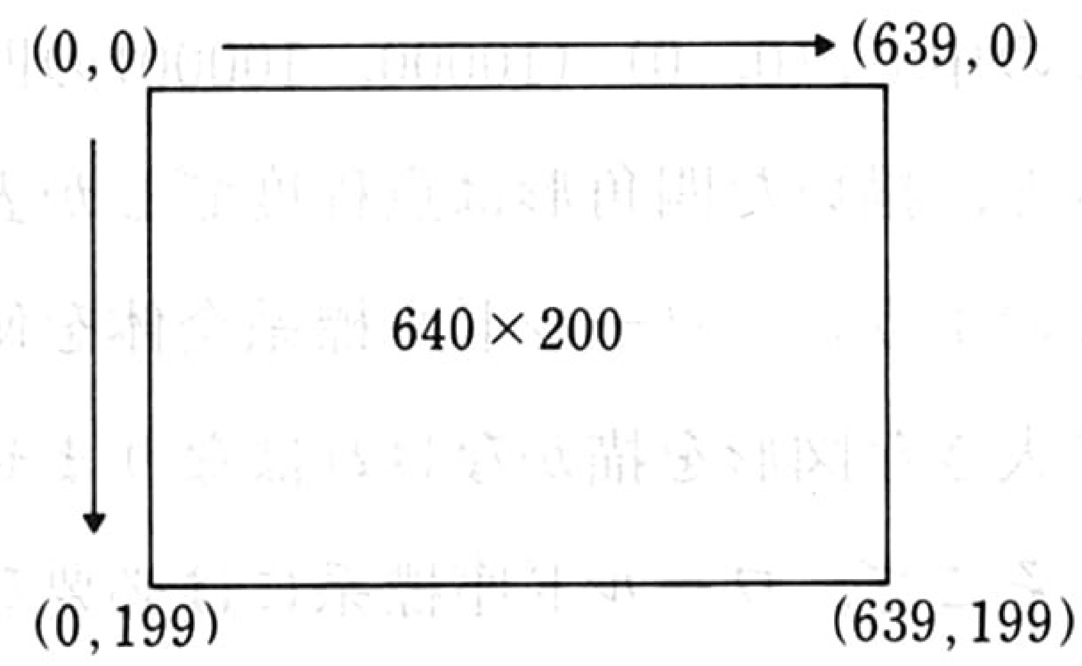
N_{88}DISK-BASICでの各座標系を次に示します。

＜キャラクタ座標系＞

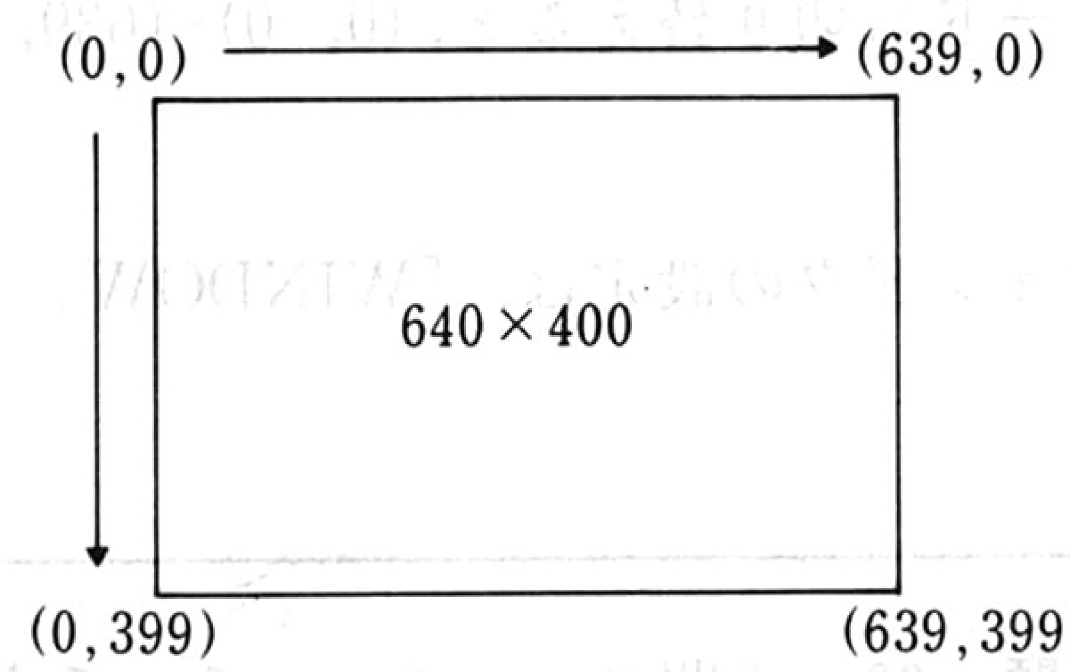


＜グラフィック座標系＞

①　白黒またはカラーモードのとき



②　高分解能白黒モードのとき



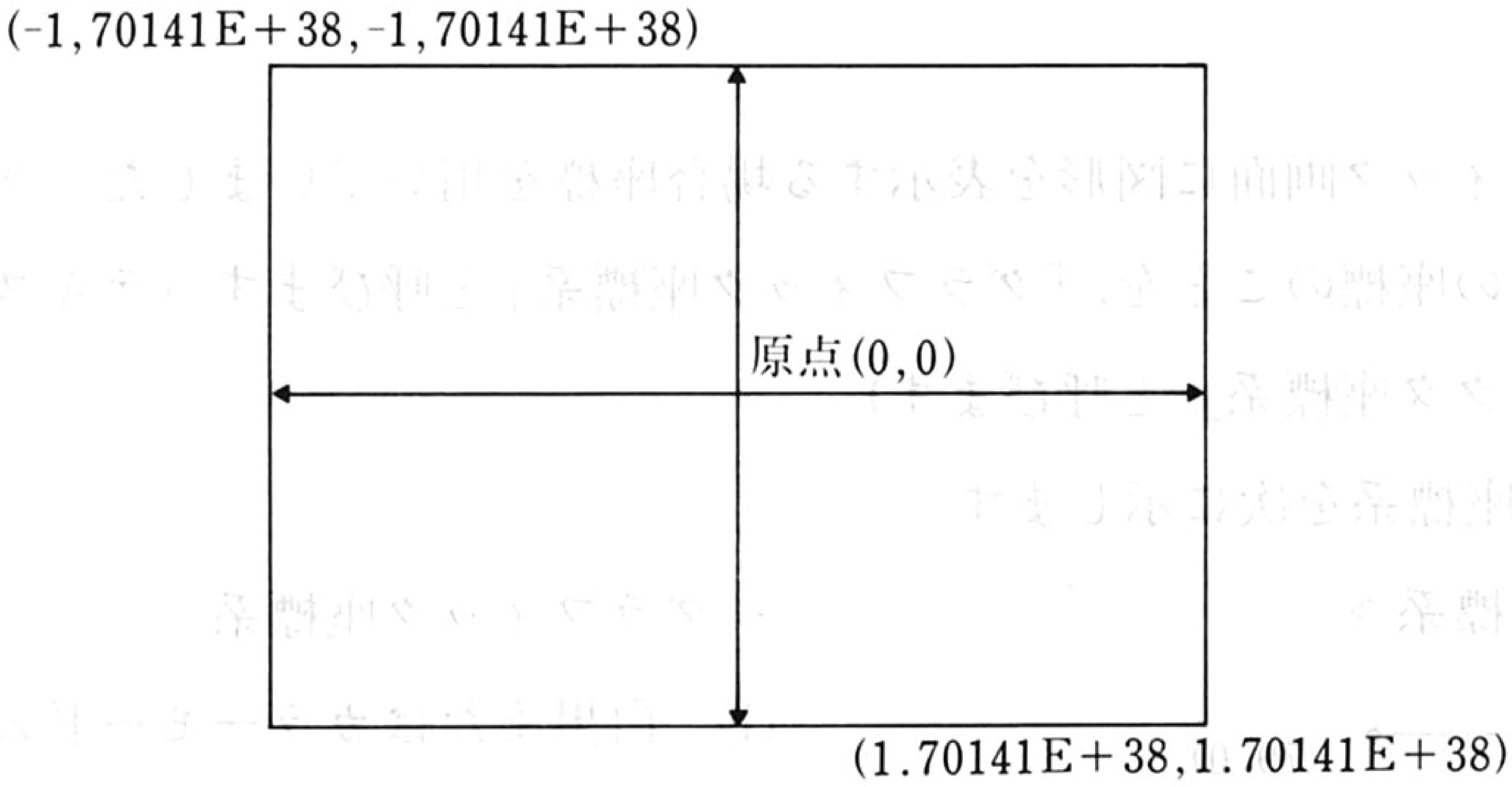
グラフィック座標系は，実際に存在する画面上の座標系(「スクリーン座標系」)と架空の大きな座標系(「ワールド座標系」)の二つを持っています。

2.5.1　ワールド座標系とは

ワールド座標系とは，N_{88}DISK-BASICが持っている架空の論理的座標のことで，グラフィック命令で描く図形はほとんどこの座標系に描かれています。この座標系は，架空のものですから，ディスプレイ画面のドット数等に捉われることなくかなり大きな座標系となります。

しかし，図形を描く座標があまりに大きいと，描くのに時間がかかったり，図形が小さく見えてしまいます。図形が小さく見えるというのは，描く座標がいくら大きくても，それを目に見えるように映し出すのはディスプレイ画面だからです。たとえば，座標(0, 0)と座標(10000, 10000)を対角とする四角形を描いたとします。表示されるディスプレイの大きさは640×200か640×400ドットです。これだけの大きさの画面に，ワールド座標系をすべて映し出した場合，ワールド座標系の大きさに対するディスプレイ画面の大きさの比率で四角形が画面に映し出されることになります。

N_{88}DISK-BASICが持っているワールド座標系は，次のようにとても大きな座標系です。



この中で，(0, 0)-(10000, 10000)の四角形を描いても，画面上にワールド座標系すべてを表示すると，描いた四角形は点程度でしか表示されません。

このように，ワールド座標系全体を使うと画面に図形を図形らしく描くにはかなり大きな座標値で大きな図形を描かなければなりません。また，図形を大きく描くと描くのに時間がかかります。そこで，ワールド座標系には必要な領域だけを使うように，「ウィンドウ」という領域を設定できるようになっています。ウィンドウというのは，画面上に表示するワールド座標系の中の領域のことで，PC-8801mkIIを起動した直後は(0, 0)-(639, 199)に設定されています。(高分解能白黒モードに切り替えると，(0, 0)-(639, 399)になります)

ウィンドウの設定は，「WINDOW」という命令で行います。

問題　22　設問に従って，いろいろなウィンドウで図形を表示して下さい。

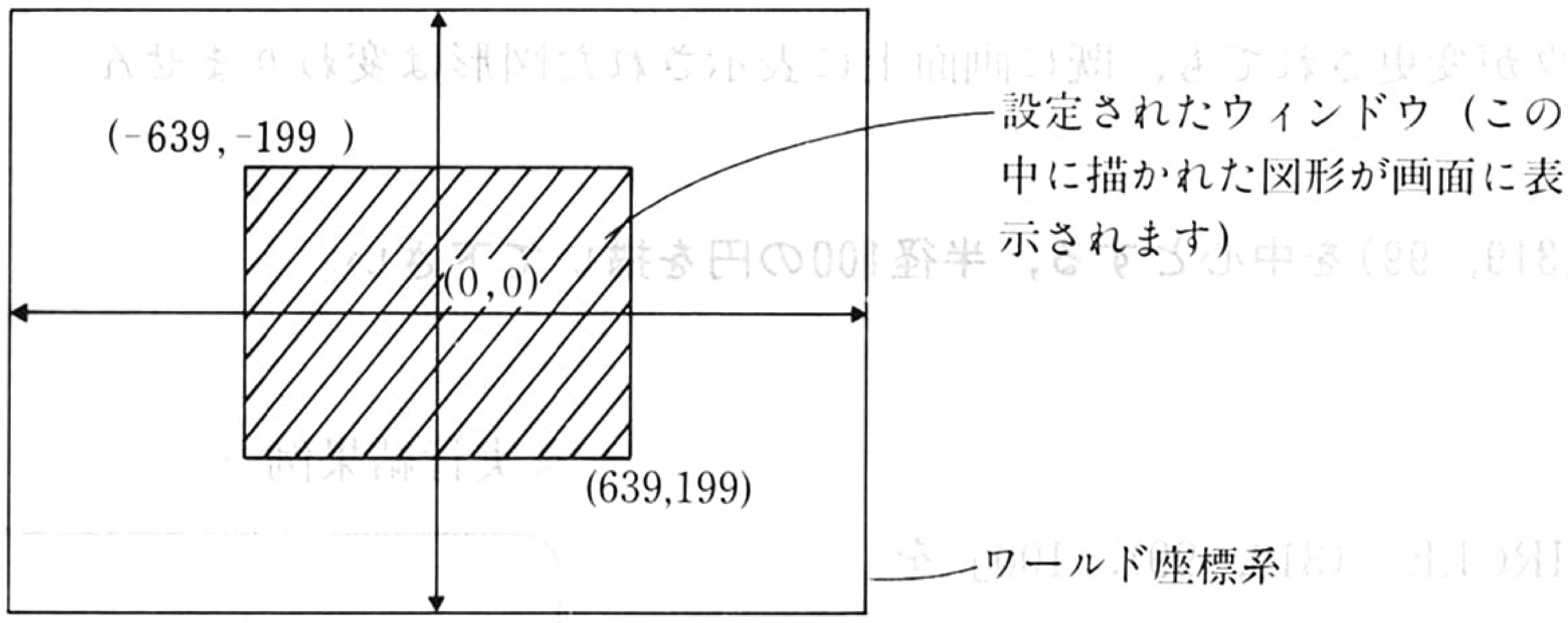
設問 22-1　グラフィック画面をクリアしてから，(-639, -199)-(639, 199)の範囲をウィンドウに設定して下さい。

＜実行結果例＞

```
CLS 2
Ok
WINDOW (-639,-199)-(639,199)
Ok
```

〔実行後の説明〕

① ワールド座標上の(−639, −199)-(639, 199)の範囲がウィンドウに設定されます。

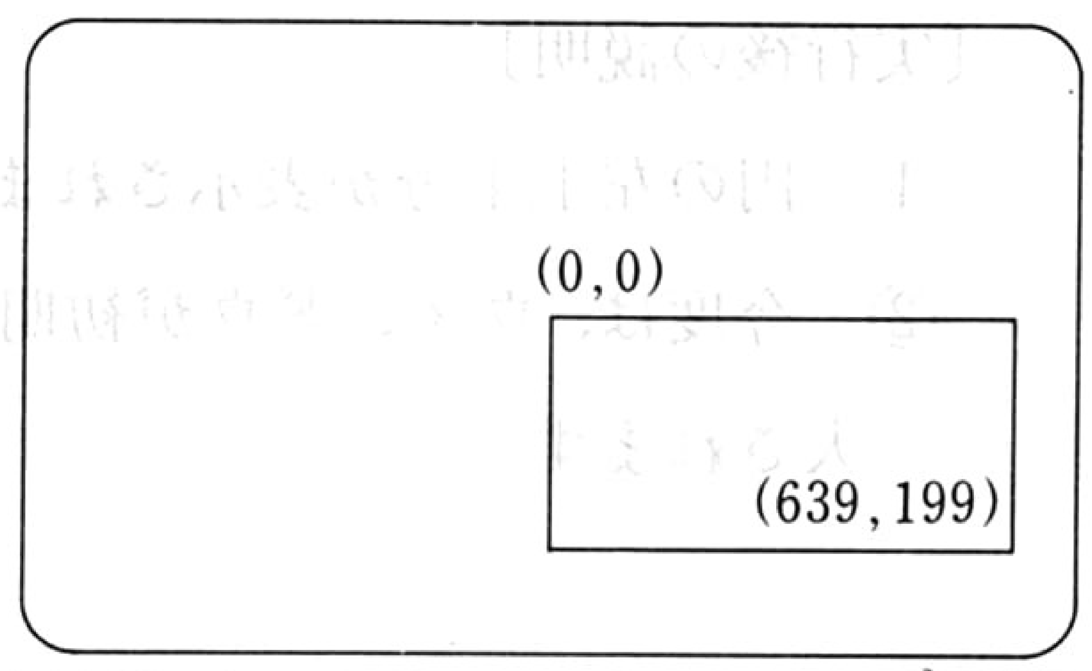


設問 22-2 (0, 0) と (639, 199) を対角とする四角形を描いて下さい。

〔操作説明〕

・命令は,「LINE (0, 0)-(639, 199),, B」をキーインします。

＜実行結果例＞



〔実行後の説明〕

① 画面の右下に四角形が表示されます。

② ウィンドウが初期値(0, 0)-(639, 199)であれば，同じLINE命令の実行で画面いっぱいに四角形が描かれます。ここでは，ウィンドウの大きさを初期状態の4倍にしたために，同じ座標値で描いた図形は4分の1に縮小されることになります。

設問 22-3 ウィンドウを(0, 0)-(319, 99)の範囲に設定して下さい。

＜実行結果例＞

```
WINDOW (0,0)-(319,99)
Ok
```

〔実行後の説明〕

① ウィンドウが(−639, −199)-(639, 199)の範囲から(0, 0)-(319, 99)の範囲に変更されます。

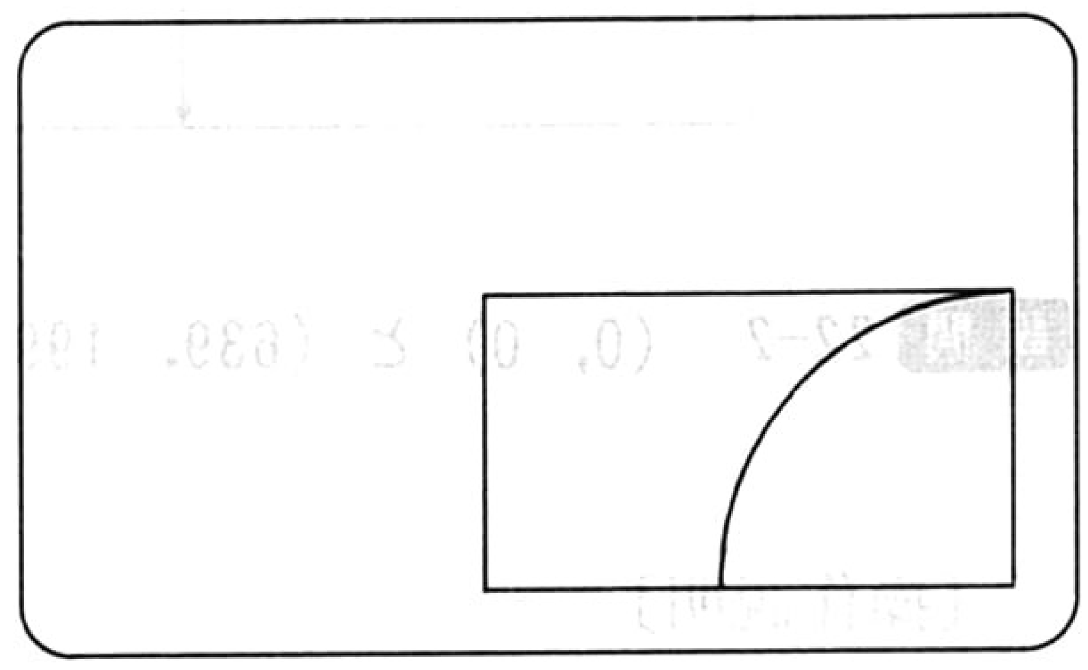
② ウィンドウが変更されても，既に画面上に表示された図形は変わりません。

設問 22-4 座標(319, 99)を中心とする，半径100の円を描いて下さい。

〔操作説明〕

・命令は，「CIRCLE (319, 99), 100」をキーインします。

＜実行結果例＞



〔実行後の説明〕

① 円の左上半分が表示されます。

② 今度は，ウィンドウが初期状態の4分の1に設定されているために，描いた図形は4倍に拡大されます。

設問 22-5 ウィンドウを初期状態(0, 0)-(639, 199)に戻して下さい。

＜実行結果例＞

```
WINDOW (0,0)-(639,199)
Ok
```

〔実行後の説明〕

① ウィンドウが初期状態に設定されます。

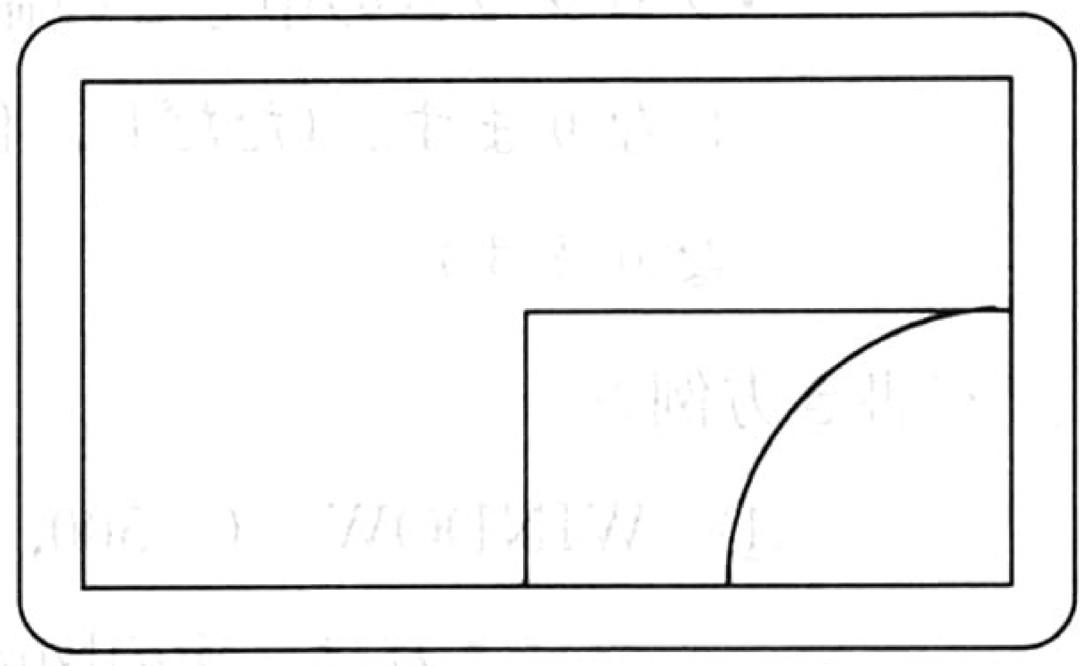
② プログラムの中でウィンドウを変更した場合は，プログラムの終わりに必ずこの命令を実行するようにします。

設問 22-6 設問22-2で描いた四角形と同じく，(0, 0)と(639, 199)を対角とする四角形を描いて下さい。

〔操作説明〕

・命令は,「LINE　(0, 0)-(639, 199), , B」をキーインします。

<実行結果例>



〔実行後の説明〕

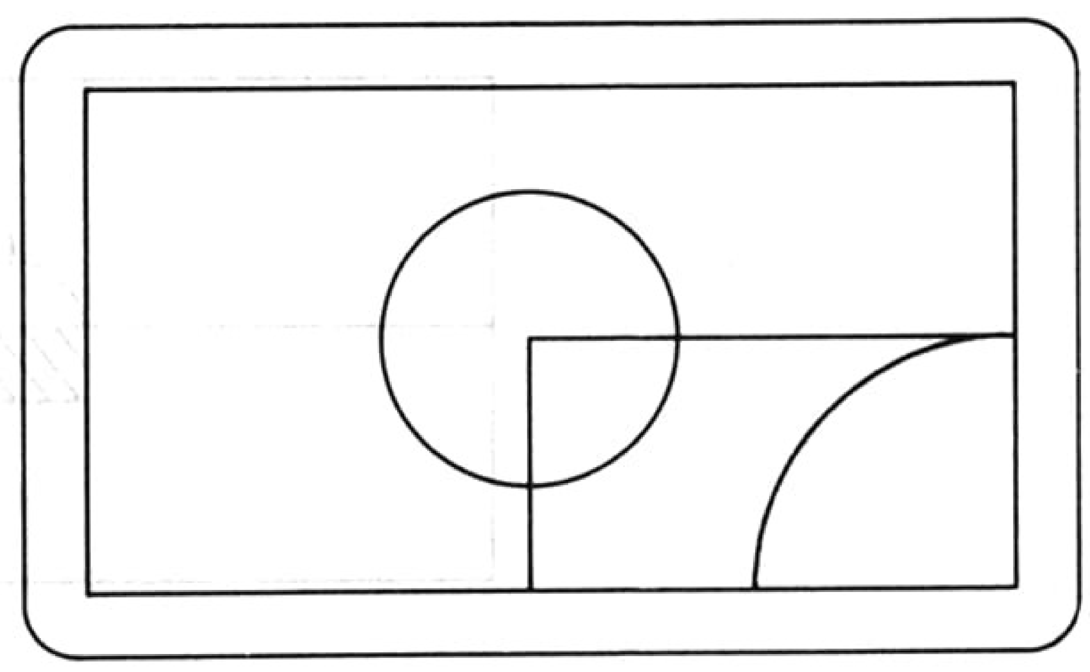
① 設問22-2と同じLINE命令を実行しても，ウィンドウが初期状態のために画面いっぱいの四角形が表示されます。

設問 22-7　設問22-4と同じように再び，座標(319, 99) を中心とする半径100の円を描いて下さい。

〔操作説明〕

・命令は,「CIRCLE　(319, 99), 100」をキーインします。

<実行結果例>



〔実行後の説明〕

① 画面の中央に，半径100の正円が表示されます。

② WINDOWの設定範囲が変わったので，設問22-4と全く同じ命令を実行しても，このように異なる図形が描かれるわけです。

【 命令のまとめ 】

WINDOW命令の一般形式と書き方例を次に示します。

<一般形式>

WINDOW　(X_1座標, Y_1座標)-(X_2座標, Y_2座標)

・ワールド座標系の座標(X_1, Y_1)の点と座標(X_2, Y_2)の点を対角線の頂点とする四角形の範囲内をウィンドウに設定します。

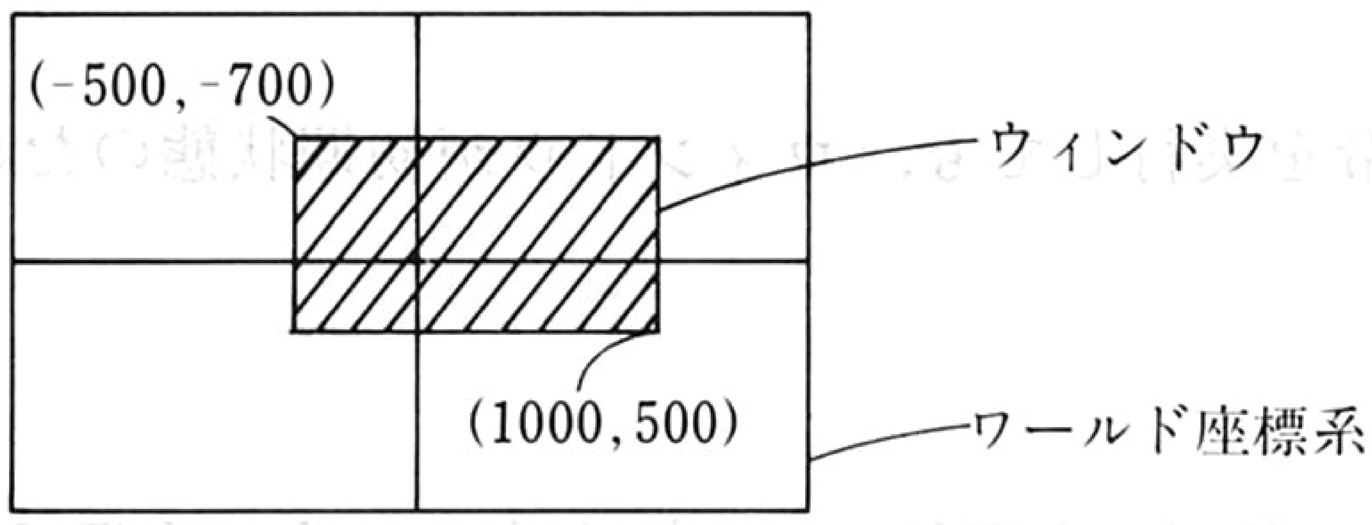
・ウィンドウの座標の範囲は，X座標，Y座標ともに−1.70141E+38～+1.70141E+38の範囲を超えてはいけません。

・プログラムの中でこの命令を省略したときは，座標(0, 0)-(639, 199)がウィンドウになります。(ただし，他のプログラムで変更してあったときは，そのときの範囲になります)

＜書き方例＞

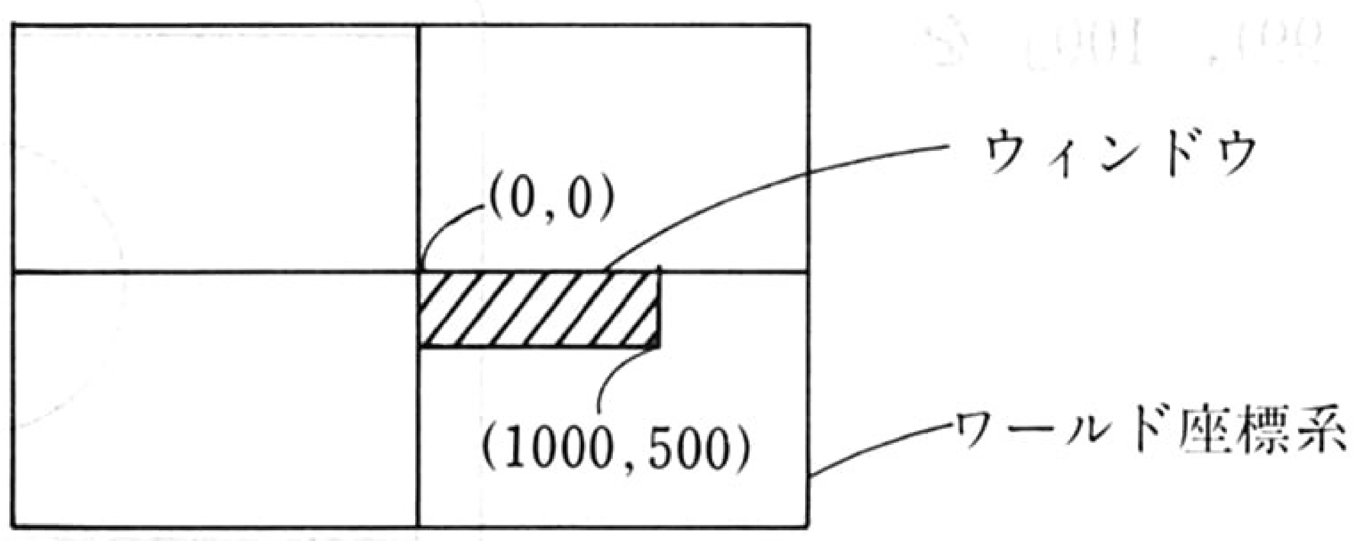
① WINDOW (−500, −700)-(1000, 50)

これは，次の図のような範囲をウィンドウとして設定する例です。



② WINDOW (0, 0)-(1000, 50)

これは，次の図のような範囲をウィンドウとして設定する例です。

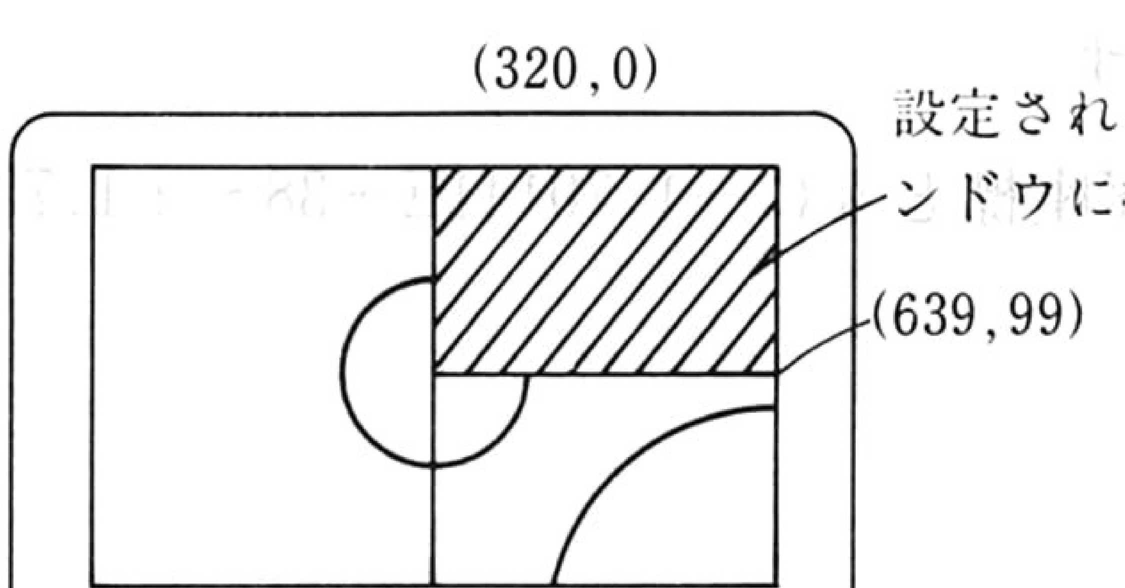


2.5.2 スクリーン座標系とは

スクリーン座標とは，ディスプレイ画面のドットに対応した物理的座標のことです。したがって，この座標系は(0, 0)-(639, 199)の大きさか(0, 0)-(639, 399)の大きさになります。

スクリーン座標系にも，ワールド座標系のウィンドウに対応するものとして「ビューポート」というものがあります。これは，設定されたウィンドウの範囲内に描かれた図形を，画面上のどの範囲に表示するかを設定するものです。たとえば，ビューポートを(320, 0)-(639, 99)に設定すると，以後ウィンドウ内に描かれる図形はディスプレイ画面の座標(320, 0)-(639, 99)に表示されることになります。

ビューポートの設定は「VIEW」という命令で行います。



ビューポートが設定されると，ウィンドウ内に描かれた図形がビューポートの大きさに縮小または拡大されて表示されます。

グラフィック命令では，このスクリーン座標系を使って図形を描いたりする命令はありませんが，次の三つの命令だけはスクリーン座標系を使用します。

・GET@命令

ビューポート内に表示されている図形を，プログラム内の配列変数に記憶する命令です。記憶する範囲指定にスクリーン座標系を使います。

・PUT@命令

GET@命令で配列変数に記憶した図形や漢字をビューポート内に表示する命令です。表示する図形や漢字の位置指定にスクリーン座標系を使います。この場合，ビューポートの頂点を原点(0，0)として扱います。

・POINT命令

相対座標の基点であるＬＰの位置を設定または変更するときに使います。

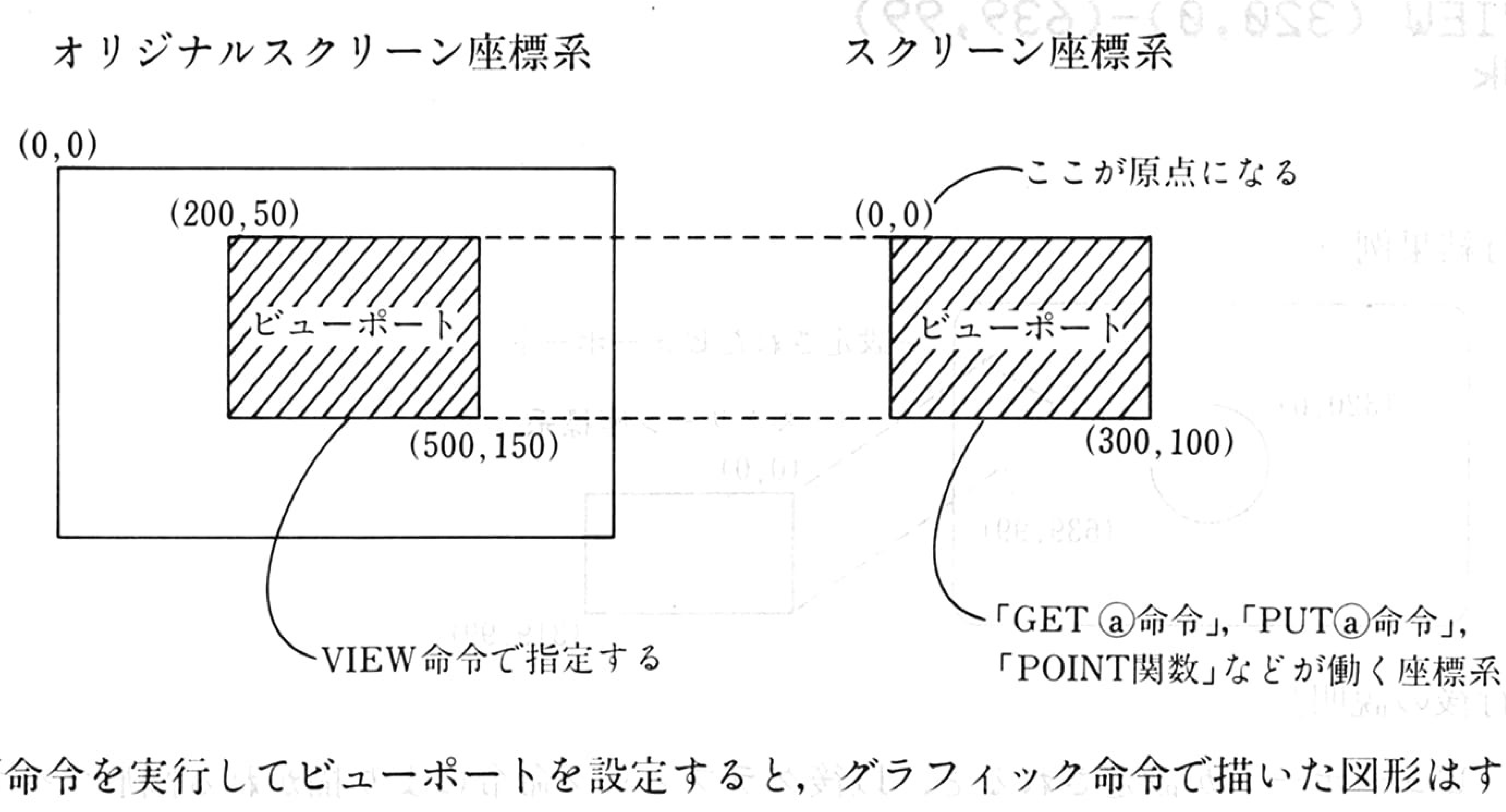
・POINT関数(１)

スクリーン座標の指定した位置に表示されているドットの色を調べるときに使います。

・POINT関数(２)

現在のＬＰの座標をスクリーン座標系またはワールド座標系の値で求めるときに使います。

これらの命令および関数は，設定されたビューポート内のスクリーン座標に対して働くものですが，ビューポートを設定するVIEW命令は，ビューポート内に展開されたスクリーン座標に対して働くのではなく，あくまでもディスプレイ画面の物理的なドットに対応したスクリーン座標(これを，特に「オリジナルスクリーン座標」と呼びます)に対して働きます。



VIEW命令を実行してビューポートを設定すると，グラフィック命令で描いた図形はすべてその内側に表示されるようになり，外側のグラフィック画面には表示されません。つまり，ビューポートの設定は，グラフィック画面を表示するディスプレイ画面の大きさおよび位置を設定することと同じであるといえます。

問題　23　設問に従って，いろいろなビューポートに図形を表示して下さい。

設問 23-1　画面をクリアしてから，座標(319，99)を中心とする半径100の円を描いて下さい。

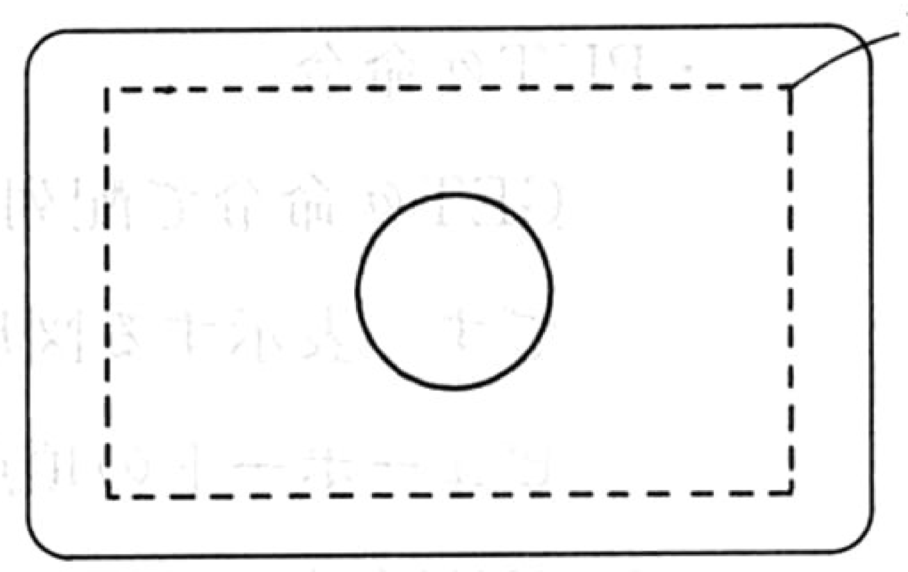
〔操作説明〕

・命令は，次のようにキーインします。

「CLS　3」

「CIRCLE　(319，99)，100」

＜実行結果例＞



〔実行後の説明〕

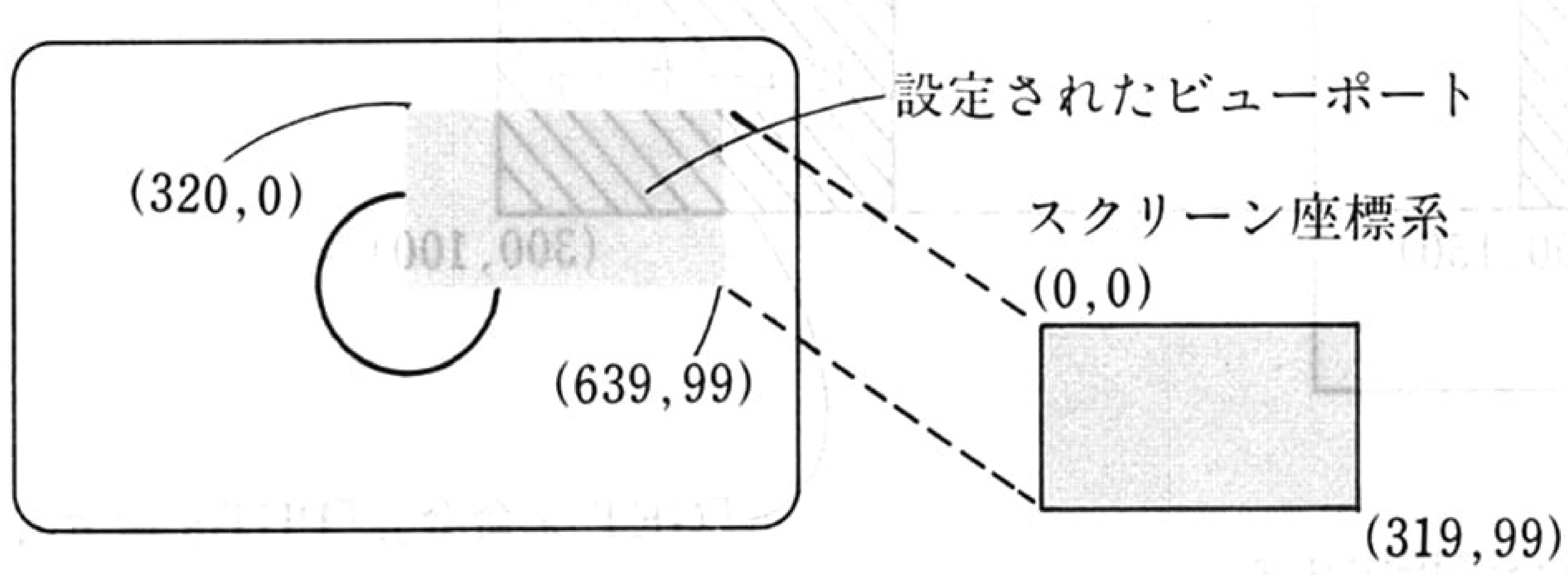
① 現在，ビューポートとウィンドウはともに初期状態(0，0)-(639，199)になっています。したがって，指定した座標の位置と大きさはディスプレイ画面の物理的な座標(ドットの配置)に一致しています。

設問 23-2　次の命令を実行して，ビューポートを画面の右上半分(320，0)-(639，99)の範囲に設定して下さい。

＜実行する命令＞

```
VIEW (320,0)-(639,99)
Ok
```

＜実行結果例＞



〔実行後の説明〕

① ビューポートが設定されると，以後グラフィック命令により描かれる図形すべてその中に表示されるようになります。

② ここで設定したビューポートは前の状態の約1/4の大きさになります。したがって，以前と同じ座標で図形を描くと1/4の大きさに縮小されて表示されます。

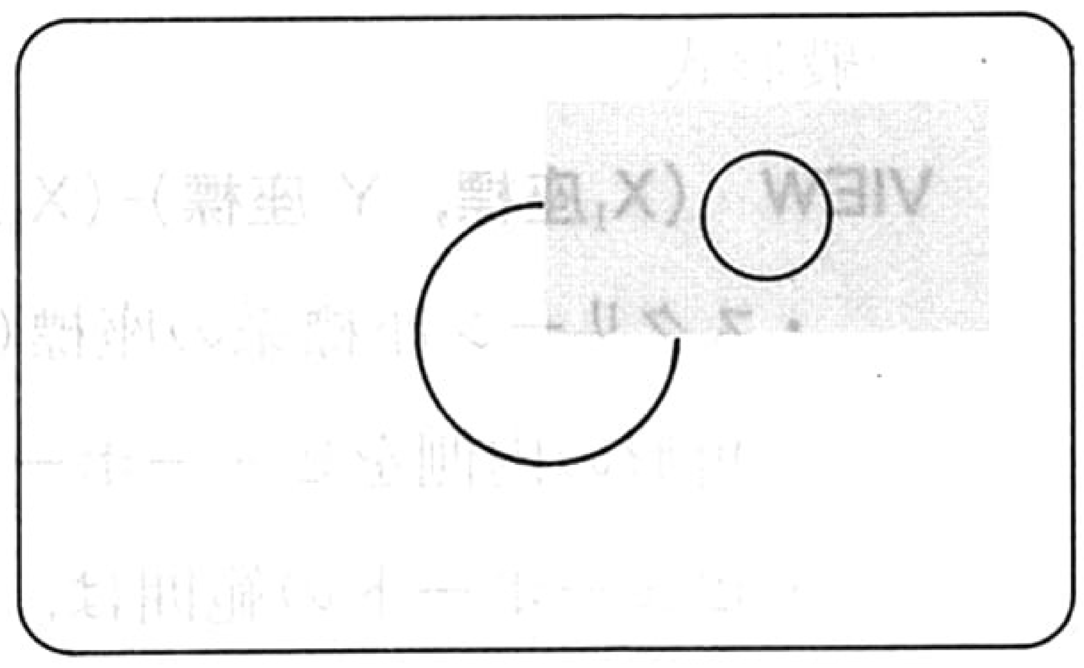
設問 23-3　設問23-1で描いた円と同じ条件の円を描いて下さい。

〔操作説明〕

・まず「CLS　3」の命令で画面をクリアします。

・命令は，「CIRCLE　(319, 99), 100」をキーインします。

＜実行結果例＞



〔実行後の説明〕

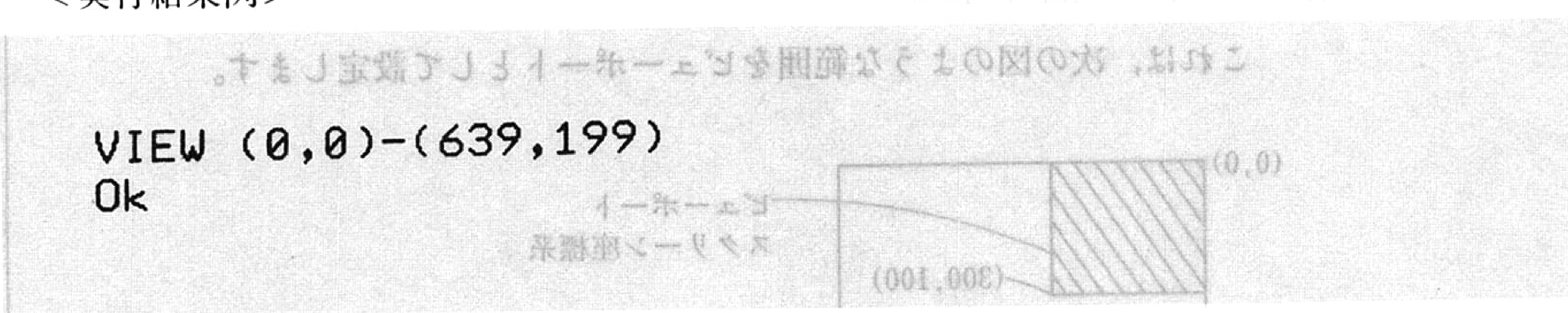
① ビューポートは，画面の右上半分に設定されています。したがって，「CLS　3」命令でクリアされるのはこの部分だけになります。

② CIRCLE命令等，図形を描く命令で指定する座標はワールド座標系になります。

③ ビューポートの設定により，同じ形の図形を自由に縮小して好きな位置に表示することができます。

設問 23-4　ビューポートを初期状態(0, 0)-(639, 199)に戻して下さい。

＜実行結果例＞



〔実行後の説明〕

① ビューポートが初期状態に設定されています。

② プログラムの中でビューポートを変更した場合は，プログラムの終わりに必ずこの命令を実行するようにします。

【 命令のまとめ 】

VIEW命令の一般形式と書き方例を次に示します。

＜一般形式＞

VIEW　(X_1座標，Y_1座標)-(X_2座標，Y_2座標)，領域色，境界色

・スクリーン座標系の座標(X_1，Y_1)の点と座標(X_2，Y_2)の点を対角線の頂点とする四角形の内側をビューポートに設定します。

・ビューポートの範囲は，X_1，X_2ともに「0」～「639」の範囲を超えてはいけません。また，Y_1，Y_2はともに「0」～「199」の範囲を超えてはいけません。ただし，高分解能白黒モードのときはY_1，Y_2は「0」～「399」の範囲まで使えます。

・座標の値は，$X_2 > X_1$，$Y_2 > Y_1$の条件でなければなりません。

・プログラムの中でこの命令を省略したときは，座標(0，0)-(639，199)がビューポートになります。ただし，そのプログラムを実行する前に既にビューポートが変更されているときはそのビューポートと同じになります。

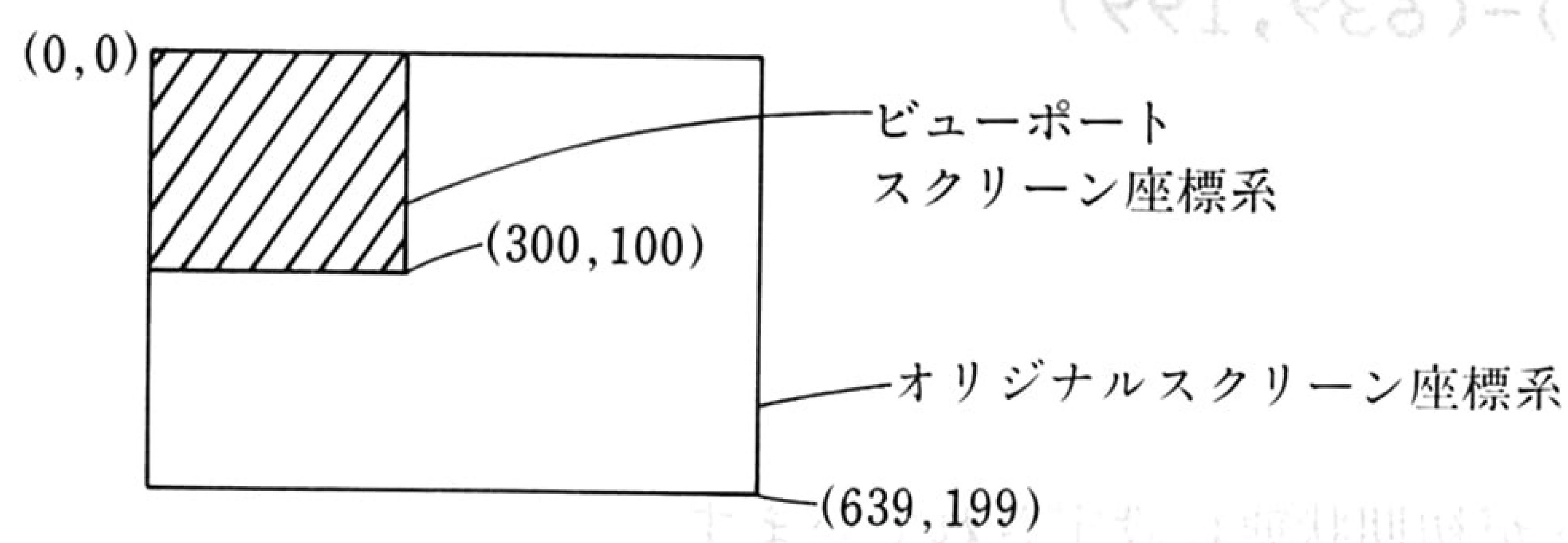
・領域色を指定すると，設定されたビューポートの範囲内がその色で塗られます。

・境界色を指定すると，設定されたビューポートの範囲の枠がその色で表示されます。

＜書き方例＞

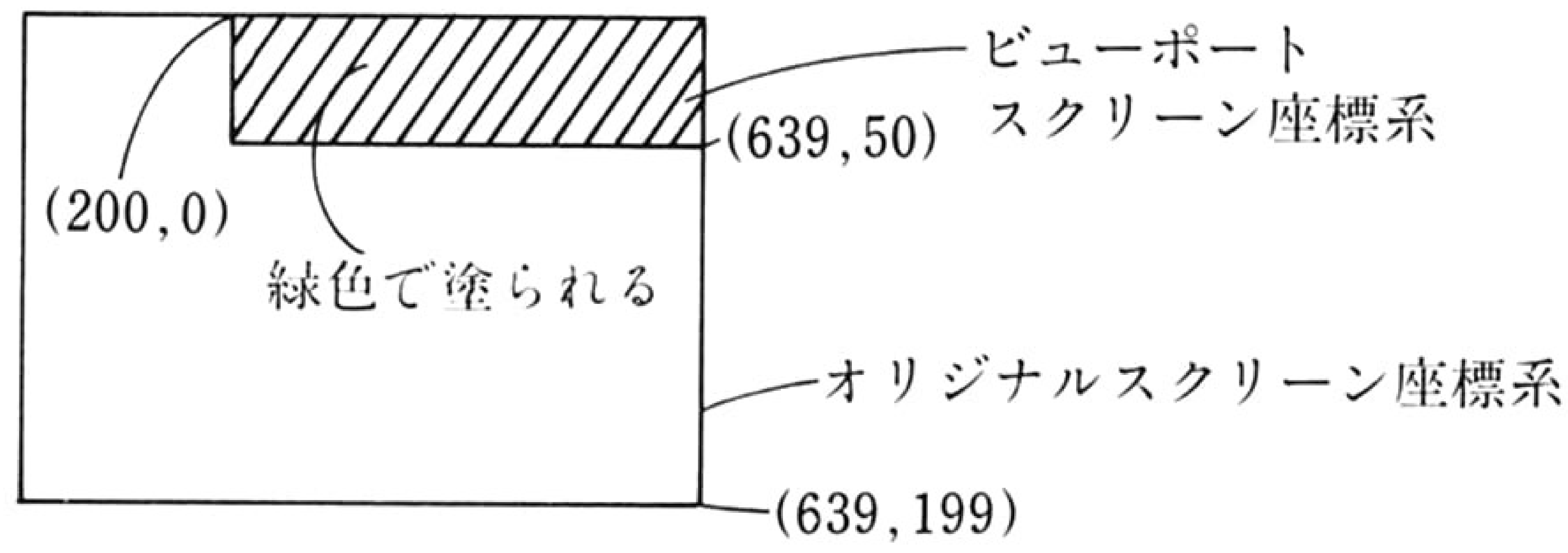
①　VIEW　(0，0)-(300，100)

これは，次の図のような範囲をビューポートとして設定します。



②　VIEW　(200，0)-(639，50)，4

これは，次の図のような範囲をビューポートとして設定した後，その範囲の内側を領域色として指定した色(ここでは緑)で塗ります。



2.5.3 図形の縮小と拡大

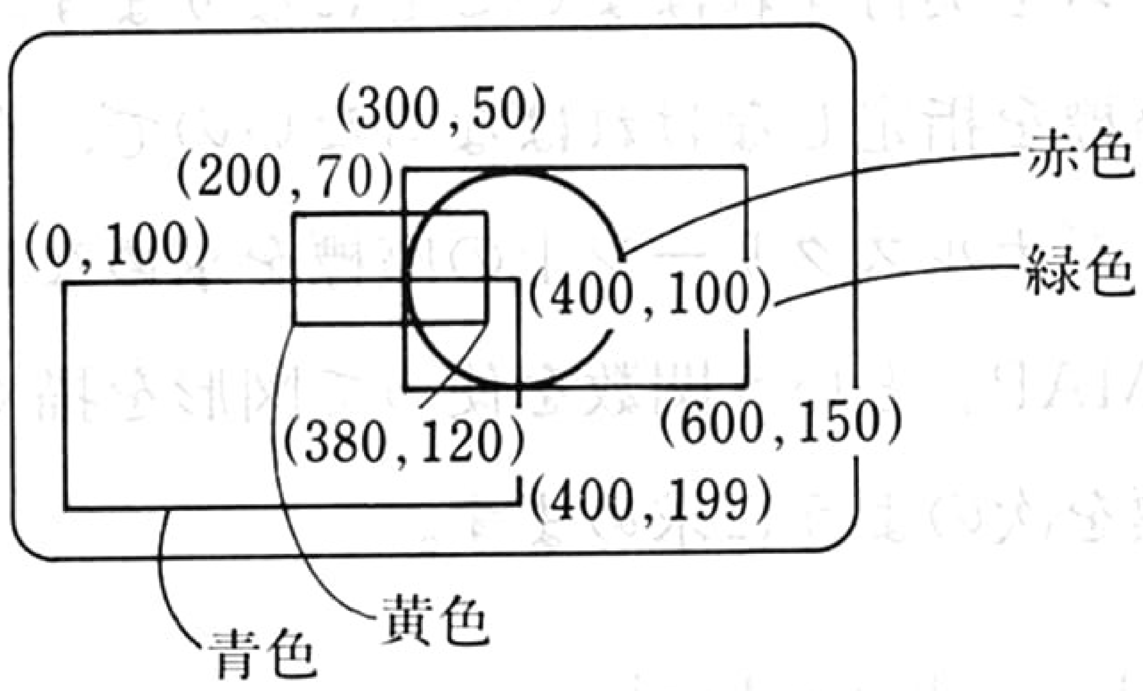
グラフィック命令で描く図形は，前述の「ウィンドウ」や「ビューポート」の設定により自由に縮小とたり拡大したりすることができます。ここでは，いろいろなウィンドウやビューポートを設定して，図形がどのように縮小あるいは拡大されるかを学習します。

問題 24 次の命令を実行すると，下のような図形が表示されます。各設問に従って，いろいろな「ウィンドウ」や「ビューポート」を設定して，図形を縮小したり拡大したりして表示して下さい。

<図形を描く命令>

```
10 CLS 3
20 LINE (0, 100)-(400, 199), 1, B
30 CIRCLE (400, 100), 100, 2
40 LINE (300, 50)-(600, 150), 4, B
50 LINE (200, 70)-(380, 120), 6, B
```

<表示される図形>



設問 24-1 画面とメモリーをクリアしてから，問題24の命令をキーインした後，これを実行して上の図形が表示されるかどうか確認して下さい。

設問 24-2 行番号50で表示された四角形の部分に，今画面に表示されている図形を縮小して表示して下さい。

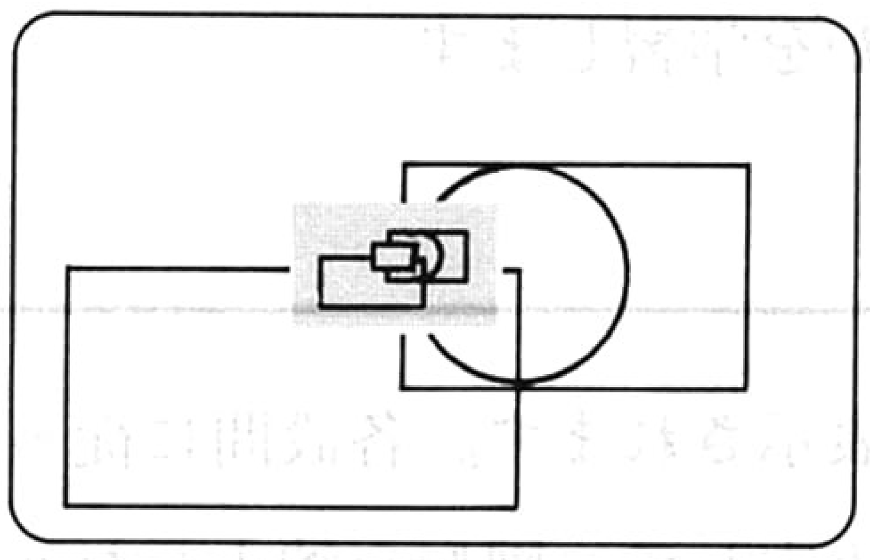
〔実行の手順〕

① 初めに，ビューポートを，行番号50で表示する四角形と同じ範囲に設定します。

```
VIEW (200,70)-(380,120)
```

② 続いて，メモリーに入っている，問題24の命令を実行して下さい。

<実行結果例>



〔実行後の説明〕

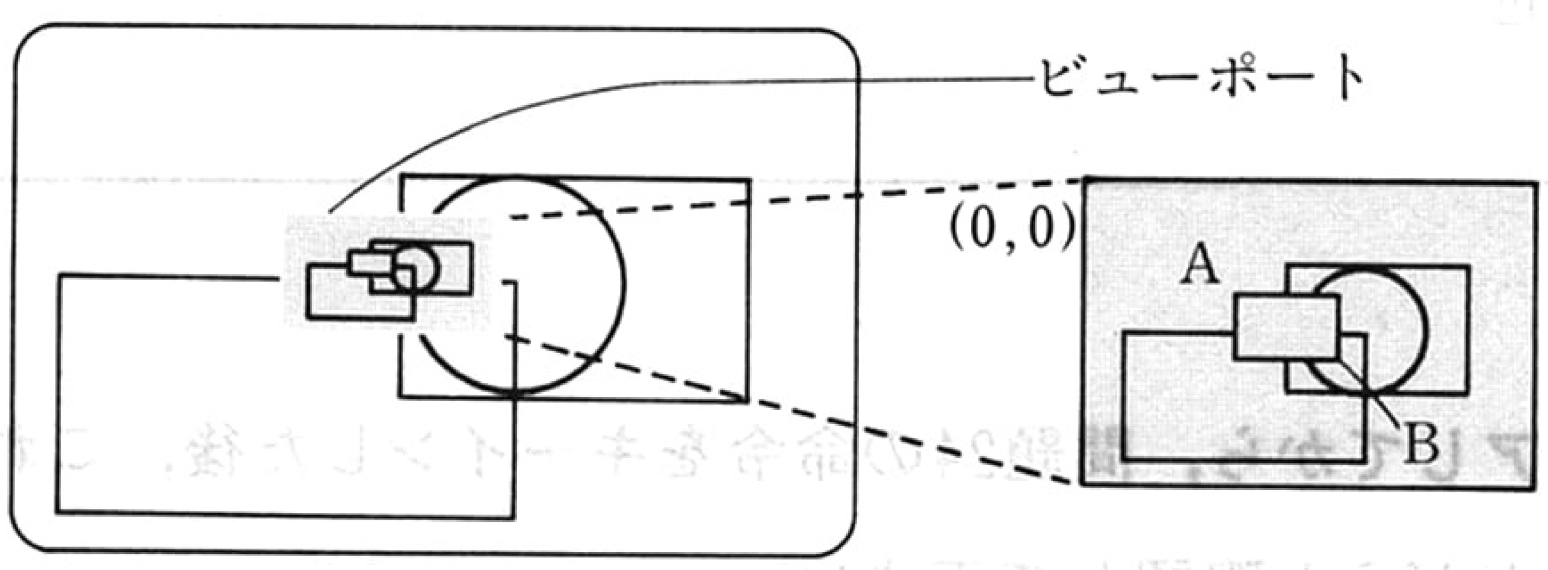
① 設定されたビューポートの内側に図形が縮小されて表示されます。

設問 24-3 設問24-2を実行したときに，行番号50で表示された四角形の中に，さらに図形を縮小して下さい。

〔実行の手順〕

① この場合も設問24-2と同様に表示された四角形の部分をビューポートに設定してから，メモリーに記憶しているプログラムを実行すればよいことになります。しかし，VIEW命令はオリジナルスクリーン上での座標を指定しなければならないので，今ビューポートとして設定しようとする四角形のオリジナルスクリーン上の座標を求めなければなりません。

そこで，このような場合は，「MAP」という関数を使って図形を描くためのワールド座標の値からビューポート内の座標を次のように求めます。



・現在設定されているビューポートは「(200, 70)-(380, 120)」なので

SX_1＝MAP(200, 0) ……… 点AのX座標

SY_1＝MAP(70, 1)………… 点AのY座標

SX_2＝MAP(380, 0) ……… 点BのX座標

SY_2＝MAP(120, 1) ……… 点BのY座標

ところが，ここで求めた点Aと点Bの座標は現在のビューポートの左上頂点を原点(0, 0)としたときの座標なので，オリジナルスクリーン座標上での点A，点Bの座標を求めるには，オリジナルスクリーン上での現在のビューポートの原点の座標値を，求めた点A，点BのX座標およびY座標のそれぞれに次のように加えます。

$OX_1 = SX_1 + 200$ ……… 点AのX座標

$OY_1 = SY_1 + 70$………… 点AのY座標

$OX_2 = SX_2 + 200$ ……… 点BのX座標

$OY_2 = SY_2 + 70$………… 点BのY座標

② このようにして求めた値を使ってビューポートを設定します。

VIEW　$(OX_1,\ OY_1)$-$(OX_2,\ OY_2)$

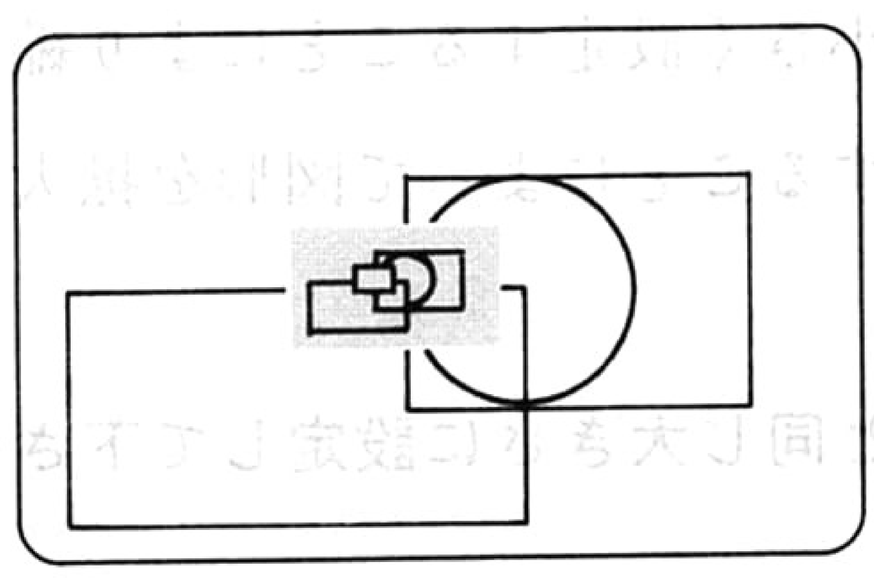
ここでは

VIEW　(256，88)-(307，100)

になります。

③ ビューポートの設定ができたら，メモリーに記憶されている，図形を描くプログラムを実行します。

＜実行結果例＞



〔実行後の説明〕

① 更に縮小された図形が表示されます。

② 更に縮小された図形の中にビューポートを設定する場合は，手順①～③と同様な操作で行います。

設問 24-4　ビューポートを初期状態(0, 0)-(639, 199)に戻して下さい。

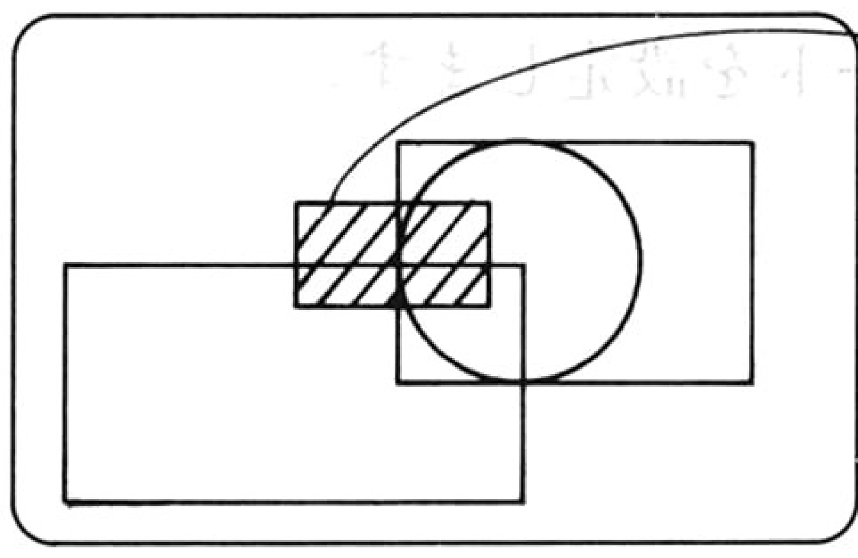
＜実行例＞

```
VIEW (0,0)-(639,199)
Ok
```

〔実行後の説明〕

① ビューポートが初期状態に戻されます。ビューポートを変更した場合は，次のプログラムを実行するまえに，必ずこの命令を実行してビューポートを初期状態にして下さい。

問題　25　問題24でキーインしたプログラムを使って，ウィンドウを変えることにより図形の拡大表示を行って下さい。



【説　明】

問題24では，図形を縮小する場合にビューポートを小さく設定することにより縮小することを説明しました。ここでは，ウィンドウの範囲を小さくすることによって図形を拡大して表示します。

設問 25-1　ウィンドウを行番号50で表示する四角形と同じ大きさに設定して下さい。

<実行例>

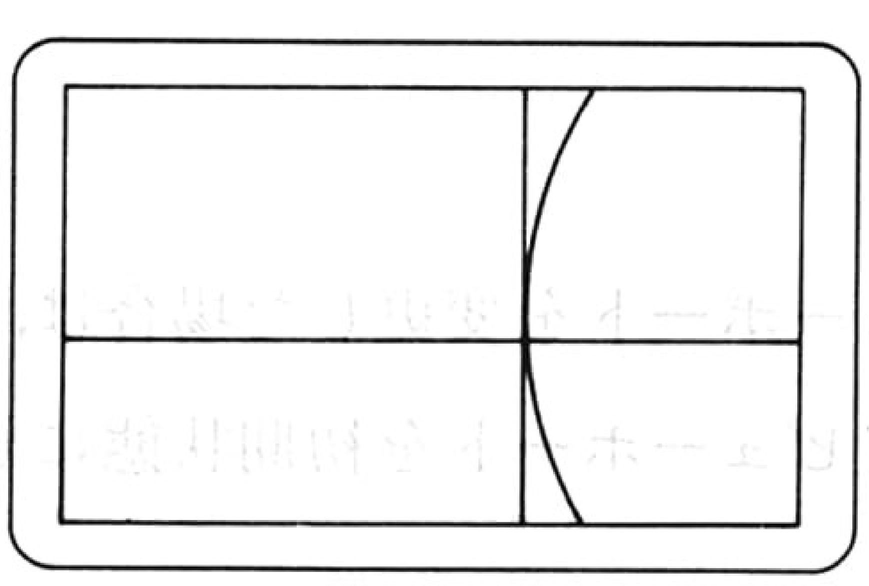
```
WINDOW (200,70)-(380,120)
Ok
```

〔実行後の説明〕

① ディスプレイ画面には，設定されたウィンドウの中に描かれた図形が現在設定されているビューポートに表示されます。

設問 25-2　メモリーに記憶されている問題24のプログラムを実行して下さい。

<実行結果例>



〔実行後の説明〕

① 行番号50で表示される四角形の内側の部分が，現在設定されているビューポートに拡大されて表示されます。

設問 25-3　ウィンドウを初期状態(0, 0)-(639, 199)に戻して下さい。

<実行例>

```
WINDOW (0,0)-(639,199)
Ok
```

〔実行後の説明〕

① ウィンドウが初期状態に戻されます。ウィンドウを変更した場合は，つぎのプログラムを実行する前に必ずこの命令を実行して，ウィンドウを初期状態にして下さい。

【 命令のまとめ 】

ワールド座標系の値から対応するスクリーン座標系の値を求めたり，逆にスクリーン座標系の値からワールド座標系の値を求めたりするMAP関数の一般形式と書き方例を次に示します。

<一般形式>

MAP(座標値，機能)

・座標値に指定されたワールド座標系またはスクリーン座標系の値から，機能の指定に従って，スクリーン座標系あるいはワールド座標系の値を求めます。

・機能は次のように指定します。

ワールド座標系の値から対応するスクリーン座標系の値を求めるとき

0　X座標の値を求めるとき

1　Y座標の値を求めるとき

スクリーン座標系の値から対応するワールド座標系の値を求めるとき

2　X座標の値を求めるとき

3　Y座標の値を求めるとき

・MAP関数で求められた値は，スクリーン座標系あるいはワールド座標系から外れてもエラーにはなりません。

<書き方例>

① MAP(5000, 0)

これは，ワールド座標系のX座標5000が，スクリーン座標系のX座標ではどの位置になるかを求めます。

② MAP(−200, 1)

これは，ワールド座標系のY座標−200が，スクリーン座標系のY座標ではどの位置になるかを求めます。

③ MAP(320, 2)

これは，スクリーン座標系のX座標320が，ワールド座標系のX座標ではどの位置になるかを求めます。

④ MAP(50, 3)

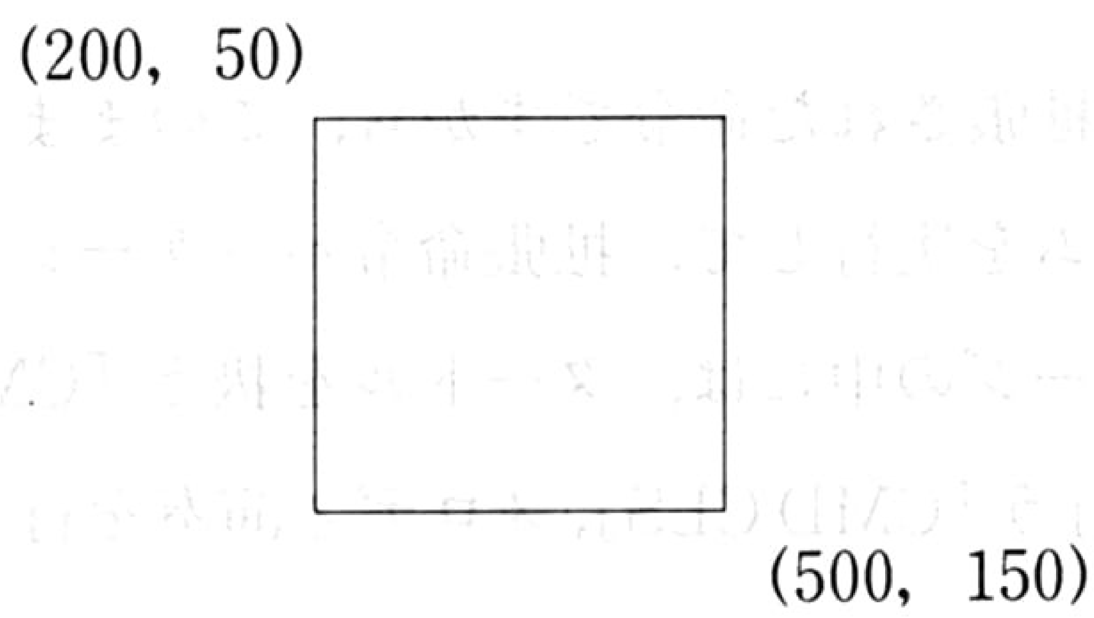
これは，スクリーン座標系Y座標50が，ワールド座標系のY座標ではどの位置になるかを求めます。

2.6 タートルの動かし方

これまでの説明で，画面に図形やグラフ等を描くときは，LINE命令やCIRCLE命令でその位置や大きさを指定して行うということはわかったと思います。ところで，PC-8801mkIIでは，これらの命令に加えて「CMD TURTLE」というタートルグラフィックを扱う命令があります。これは，画面上に「タートル(亀)」と呼ばれる三角形の図形を表示して，「前へ進め」とか「右へ曲がれ」とかの指示を与えてタートルを動かし，その軌跡で図形を表示するものです。

ここでは，タートルを使っていろいろな図形を描く方法を学習します。

問題 26 タートルを使って，次のような四角形を描いて下さい。



【説　明】

タートルは，命令の指示によって画面上を自由に動き回ります。タートルを動す「CMD TURTLE命令」は，「タートルの表示と消去」，「軌跡の表示と消去」，「進行方向の指定(前，後，左，右)や角度」，「移動距離」等を指定します。

タートルの表示は，現在のLP(Last referenced Point：「2.4　絶対座標と相対座標」参照)の位置に行われます。PC-8801mkIIを起動した直後や画面をクリアした後の，LPは(0, 0)の位置にあります。したがって，いきなりタートルを表示すると，タートルは画面の左上隅に一部分だけ表示されることになります。そこで，上の図を描く場合はまずPOINT命令でLPを座標(200, 50)の位置に移動します。その後，右へ300ドット，下へ100ドット，左へ300ドット，上へ100ドットという具合にタートルを移動して四角形を描きます。

設問 26-1 画面をクリアしてからLPを座標(200, 50)に移動させて下さい。

<実行例>

```
CLS 3
        ←ここで画面がクリアされる
Ok
POINT (200,50)
Ok
```

〔実行後の説明〕

① LPが座標(200, 50)に移動します。

設問 26-2 タートルを扱う命令「CMD TURTLE」が使えるように，N_{88}-BASICの拡張命令パッケージを追加して下さい。

【説 明】

CMD TURTLE命令は，N_{88}-BASICに拡張された命令ですから，このままでは実行できません。そこで，「@load.n88」というプログラムを実行して，拡張命令パッケージをメモリーに展開しておく必要があります。拡張命令パッケージの中には，タートルを扱う「CMD TURTLE」のほかに，グラフィック画面の高度クリアを行う「CMD CLS」，メロディ演奏を行う「CMD SING」などの命令があります。

〔実行の手順〕

① システムディスクの中から「@load.n88」というプログラムをロードして下さい。

```
load "@load.n88"
Ok
```

② プログラムを実行して下さい。

```
run
The following statements are available.

    1. CMD TURTLE
    2. CMD SING
    3. CMD CLS
    4. CMD TEXT ON/OFF
    5. CMD CUT

Please enjoy new features.
Ok
```

〔実行後の説明〕

これで，拡張命令パッケージがメモリーに追加されたことになります。

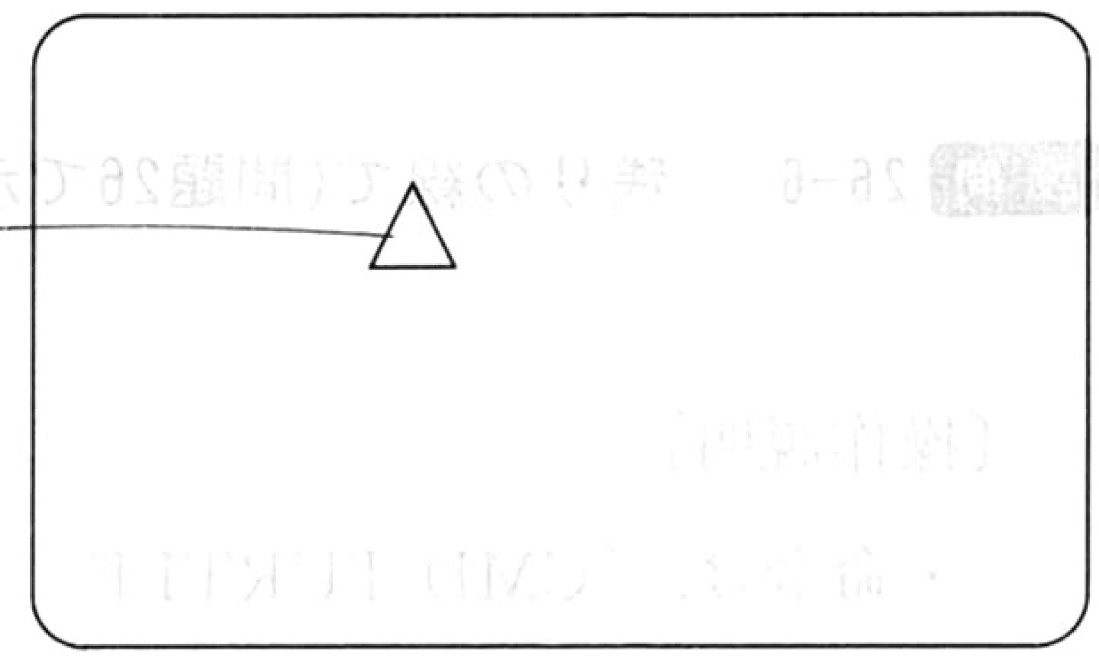
(注) 「@load.n88」というプログラムは，さらに「@exst」というマシン語のプログラムをロードして実行しています。実際の拡張命令パッケージは，この「@exst」というマシン語のプログラムの中に入っています。

設問 26-3　タートルを表示して下さい。

〔操作説明〕

・命令「CMD TURTLE　"ST"」
　をキーインします。

＜実行結果例＞



座標(200,50)の位置に上向きでタートルが表示される

〔実行後の説明〕

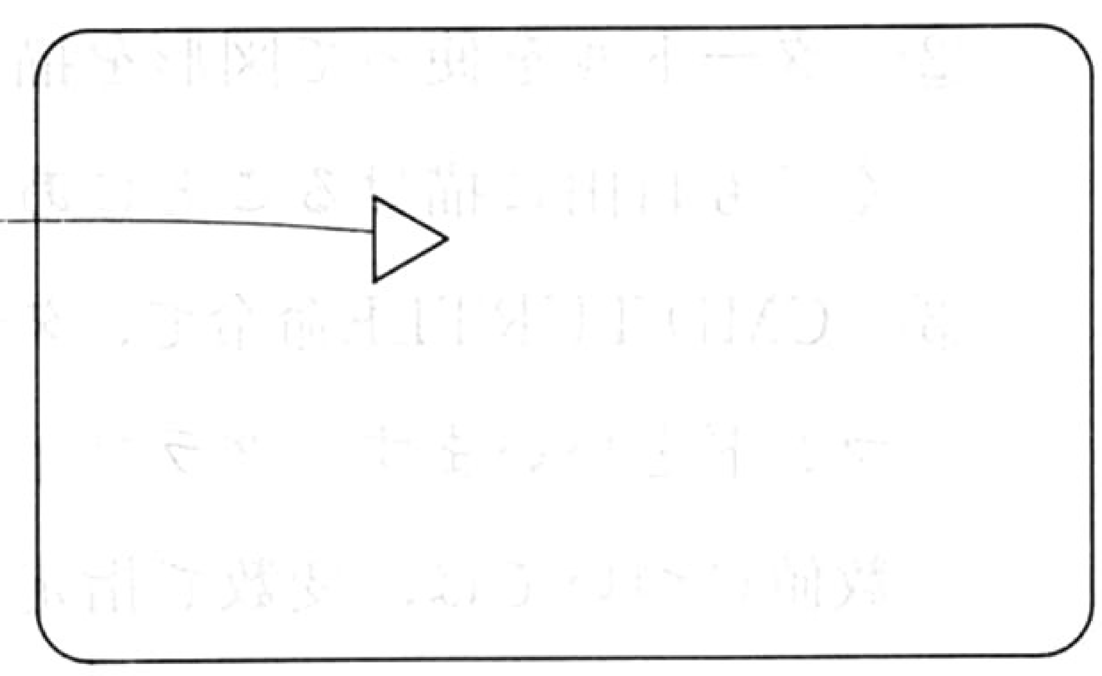
①　タートルが上向きで表示されます。

設問 26-4　タートルを右へ300ドット移動して下さい。

〔操作説明〕

①　タートルを右へ移動させるために，タートルの向きを右へ90°変更します。

・命令は，「CMD　TURTLE　"RT90"」
　をキーインします。

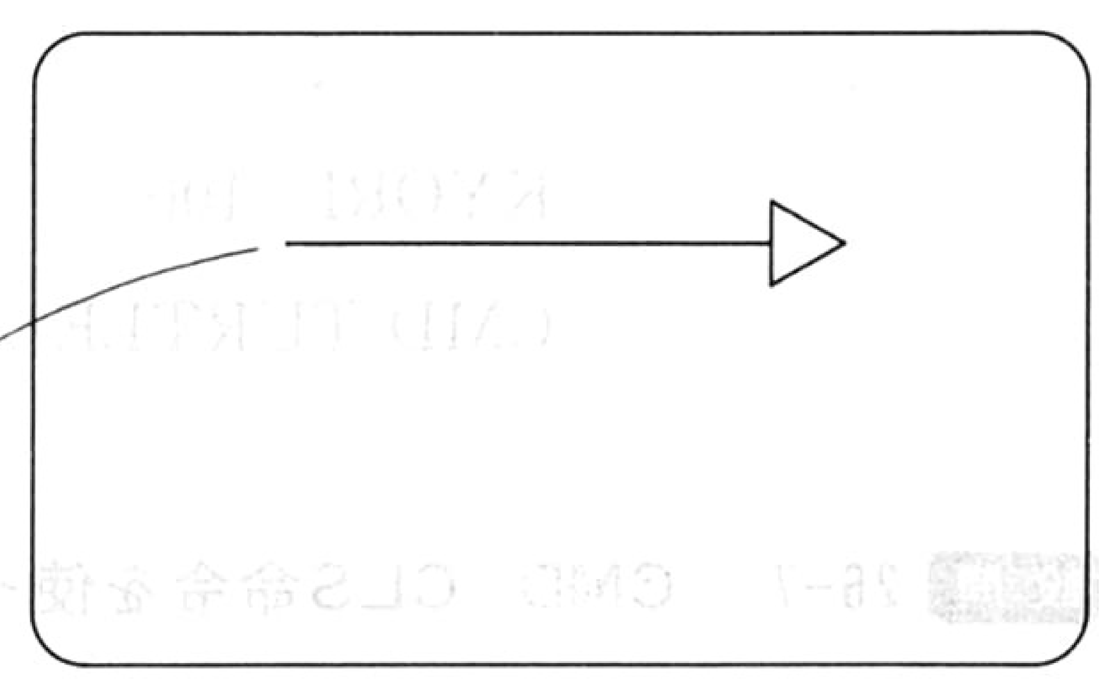


タートルが右を向く

②　タートルを300ドット進めます。

・命令は，「CMD TURTLE　"FD150"」
　をキーインします。

・水平方向へタートルを移動させるときは移動させるドット数の½の値をFDコマンドで指定します。



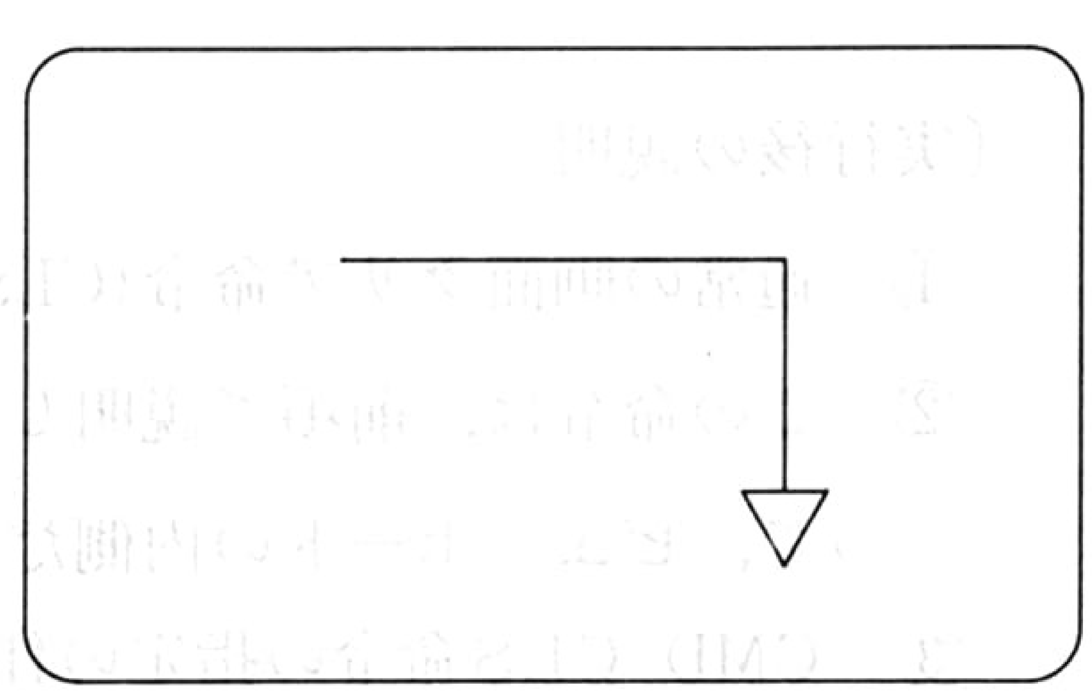
タートルが300ドット進んで，その軌跡が描かれる

設問 26-5　タートルを現在向いている方向から右へ100ドット移動して下さい。

〔操作説明〕

・命令は，「CMD TURTLE
　"RT90　FD100"」をキーインします。

＜実行結果例＞



〔実行後の説明〕

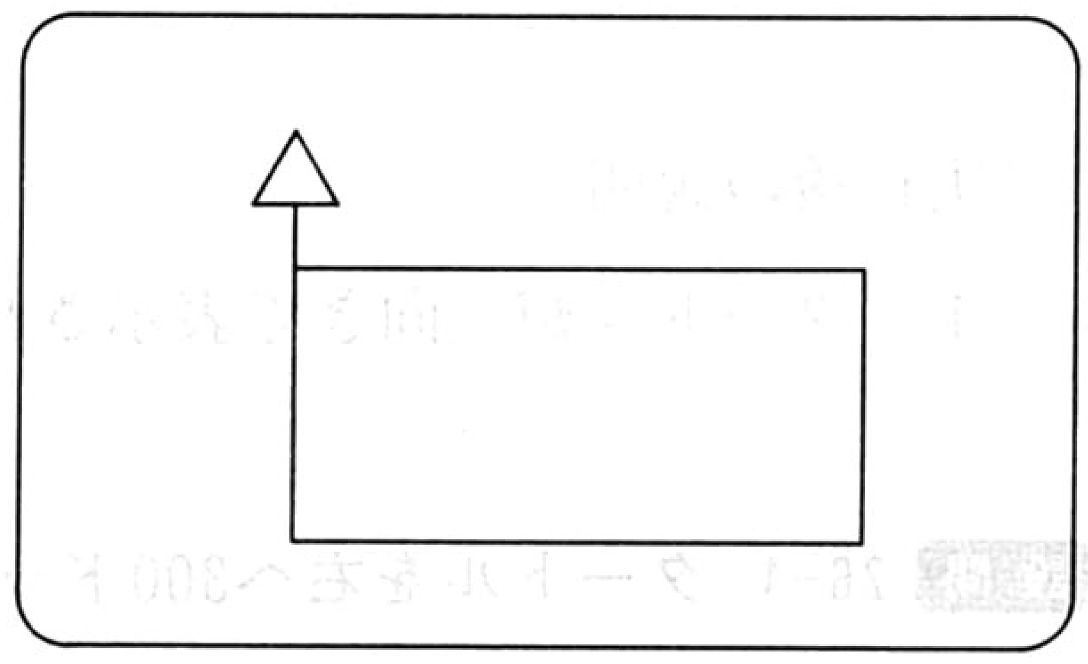
① タートルに対する複数の指示を，一つの「CMD TURTLE」命令で実行することができます。

設問 26-6　残りの線で(問題26で示された四角形の下側と左の線)を描いて下さい。

〔操作説明〕

・命令は，「CMD TURTLE
”RT90　FD150　RT90　FD100”」
をキーインします。

＜実行結果例＞



〔実行後の説明〕

① タートルを使って四角形を描くことができましたが，このように座標位置の決まっている場合の四角形を描く場合は，タートルを使うよりもLINE命令で描いた方が簡単です。

② タートルを使って図形を描くことの利点は，LINE命令などと違い，座標位置が分からなくても自由に描けることにあります。

③ CMD TURTLE命令で，タートルの動きを指示する文字列のことをグラフィックサブコマンドといいます。グラフィックサブコマンドは，文字と数値で指定しますが，このうち数値については，変数で指定しても構いません。ただし，変数名はカッコで囲みます。

(例)　CMD TURTLE　”FD100”
　　　　　↓
　　　KYORI＝100
　　　CMD TURTLE　”FD(KYORI)”

設問 26-7　CMD　CLS命令を使って，高速で画面をクリアして下さい。

＜実行結果例＞

```
CMD CLS 3   ←――ここで画面がクリアされる
Ok
```

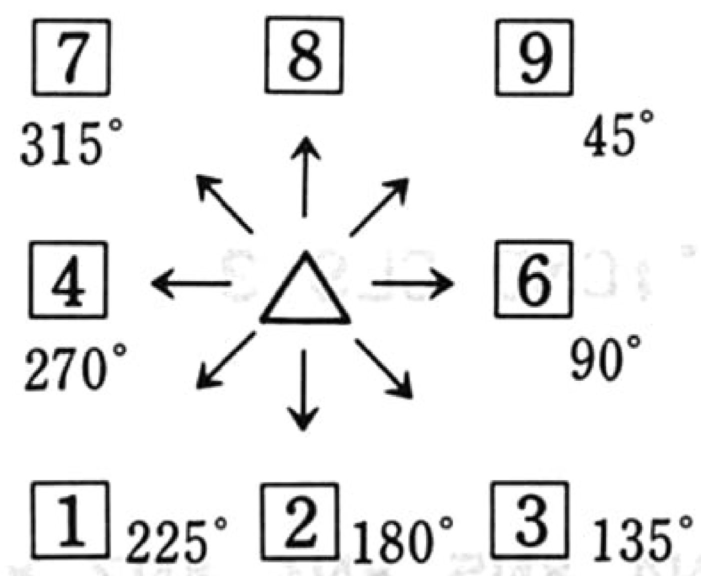
〔実行後の説明〕

① 通常の画面クリア命令(CLS命令)よりはるかに高速で画面がクリアされます。

② この命令は，前項で説明したビューポートは無視して画面全部をクリアします。したがって，ビューポートの内側だけをクリアする場合は，CLS命令で行います。

③ CMD　CLS命令の指定の仕方はCLS命令と同じなので，ここでは省略します。

問題　27　タートルの動きを，テンキーで操作できるようなプログラムを作って，画面にいろいろな図形を描いて下さい。



設問 27-1　メモリーをクリアしてから、次のプログラムをキーインして下さい。

＜キーインするプログラム＞

```
100 SCREEN 0,0:CMD TURTLE 'HT':CMD CLS 3
110 POINT (320,100)
120 CMD TURTLE 'ST'
130 N$=INPUT$(1):N%=VAL(N$)
140 ON N% GOSUB *N1,*N2,*N3,*N4,*N5,*N6,*N7,*N8,*N9
150 GOTO 130
160 *N1:CMD TURTLE 'HD225 FD1':RETURN
170 *N2:CMD TURTLE 'HD180 FD1':RETURN
180 *N3:CMD TURTLE 'HD135 FD1':RETURN
190 *N4:CMD TURTLE 'HD270 FD1':RETURN
200 *N5:RETURN
210 *N6:CMD TURTLE 'HD90 FD1':RETURN
220 *N7:CMD TURTLE 'HD315 FD1':RETURN
230 *N8:CMD TURTLE 'HD0 FD1':RETURN
240 *N9:CMD TURTLE 'HD45 FD1':RETURN
```

〔キーイン後の説明〕

① 行番号100は，カラーモードにした後，画面に高速でクリアします。タートルを使う場合は，画面をクリアする前に必ずタートルの消去(HTサブコマンド)を実行します。

② 行番号110は，LPを(320, 100)に設定します。

③ 行番号120は，タートルの表示を行います。

④ 行番号130は，数値(テンキー)の入力を行います。(数値以外は「0」になります。)

⑤ 行番号140は，入力された値にしたがって処理を行うようにしています。

⑥ 行番号150は，入力された値が，「0」のとき，何もしないで行番号130へ戻るようにしています。

⑦ 行番号160～230は，入力された値(テンキー)の方向にタートルを向けて1進めます。

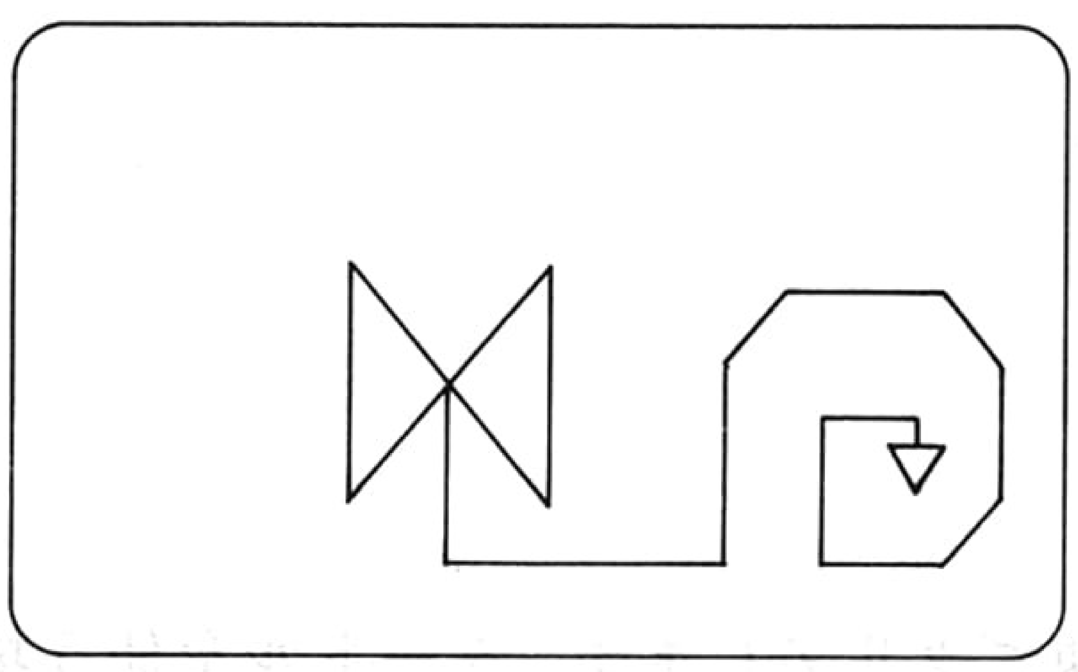
設問 27-2　プログラムのリストを画面に表示して，正しくキーインできているかどうか確認して下さい。

＜確認用リスト＞

```
100 SCREEN 0,0:CMD TURTLE "HT":CMD CLS 3
110 POINT (320,100)
120 CMD TURTLE "ST"
130 N$=INPUT$(1):N%=VAL(N$)
140 ON N% GOSUB *N1,*N2,*N3,*N4,*N5,*N6,*N7,*N8,*N9
150 GOTO 130
160 *N1:CMD TURTLE "HD225 FD1":RETURN
170 *N2:CMD TURTLE "HD180 FD1":RETURN
180 *N3:CMD TURTLE "HD135 FD1":RETURN
190 *N4:CMD TURTLE "HD270 FD1":RETURN
200 *N5:RETURN
210 *N6:CMD TURTLE "HD90 FD1":RETURN
220 *N7:CMD TURTLE "HD315 FD1":RETURN
230 *N8:CMD TURTLE "HD0 FD1":RETURN
240 *N9:CMD TURTLE "HD45 FD1":RETURN
```

設問 27-3　正しくキーインできていたら，プログラムを実行していろいろな図形を描いてみて下さい。

＜実行結果例＞



〔実行後の説明〕

① テンキーからの操作により，タートルの向きを45°単位で変更しながら図形を描くことができます。

② 実行を終了する場合は，STOP キーを押して下さい。

設問 27-4　画面をクリアしてから，このプログラムをドライブ2のフロッピーディスクへ「MONDAI.27」という名前で保存して下さい。

＜実行結果例＞

```
CMD CLS 3      ←ここで画面がクリアされる
Ok
save "2:MONDAI.27"
Ok
```

【 命令のまとめ 】

① CMD TURTLE命令の一般形式と書き方例を次に示します。

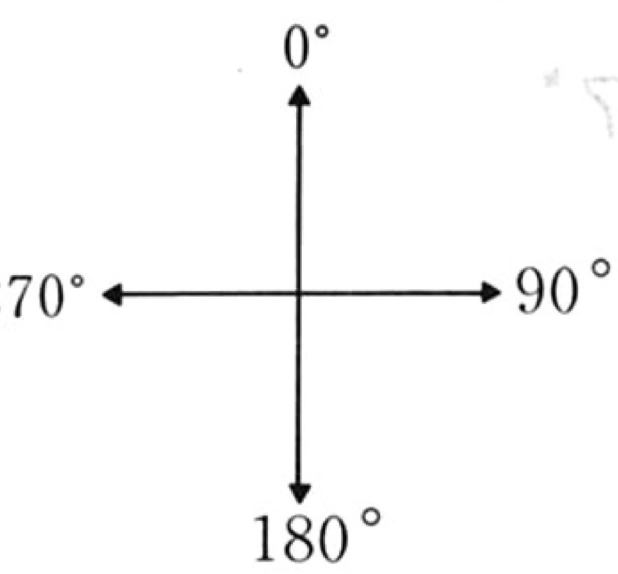
＜一般形式＞

CMD TURTLE "グラフィックサブコマンド"

・グラフィックサブコマンドの指示にしたがって，画面上のタートルを動かし，いろいろな図形を描きます。

・グラフィックサブコマンドの種類と機能を一覧表形式で示します。

種　類	条　　件	機　　　　能
ST		LPの位置にタートルを表示します。
HT		タートルを消します。
FDn	−32768≦ n ≦32767	タートルをnドット前進させます。ただし，水平方向に対しては2nドット移動します。負の値を指定すると後退します。(タートルの向きは変わりません)
BKn	−32768≦ n ≦32767	タートルをnドット後退させます。ただし，水平方向に対しては2nドット移動します。負の値を指定すると前進します。(タートルの向きは変わりません)
LTn	−32768≦ n ≦32767	タートルをn度左へ向けます。
RTn	−32768≦ n ≦32767	タートルをn度右へ向けます。
PCn	0≦ n ≦7	タートルが描く軌跡(ペン)の色を指定します。
PU		タートルのペンを上げます。軌跡は描かれません。
PD		タートルのペンを下げます。タートルは軌跡を残しながら移動します。
MVx,y	−32768≦ x, y ≦32767	*x*，*y*で指定したスクリーン座標上の位置(*x*，*y*)にタートルを移動します。
SXx	−32768≦ y ≦32767	タートルをスクリーン座標(*x*,(現在のy座標))の位置へ水平移動します。

種　類	条　　　件	機　　　　　能
SYy	−32768≦ y ≦32767	タートルをスクリーン座標((現在のx座標), y)の位置へ垂直移動します。
HDn	−32768≦ n ≦32767	nで指定された角度へタートルの向きを変えます。  正の数で指定すると，時計まわりの角度指定になります。 負の数で指定すると，時計まわりと逆方向になります。
RPj[…]	1≦ j ≦225	[…]に指定されたグラフィックサブコマンドをj回繰り返します。 […]の中に，さらにRPj[…]を指定することができます。ただし，8レベルまでです。
CPn	n＝0，1	640×200モードで，X方向の補正を行うかどうかを0(補正しない)，1(補正する)で指定します。 通常は，CP1(補正する)で使用します。

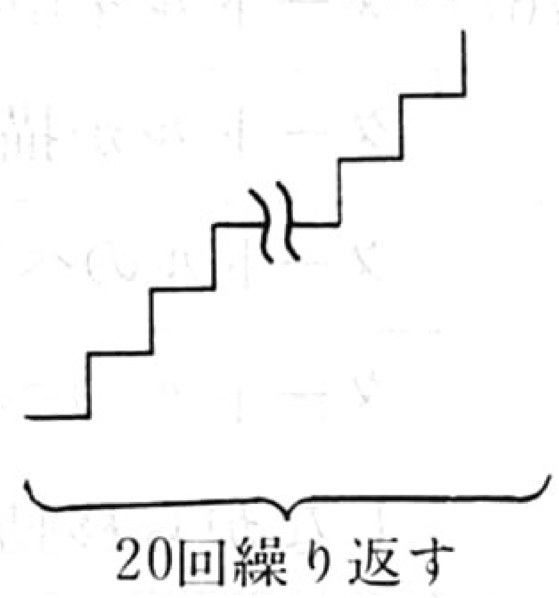
<書き方例>

① CMD TURTLE　"ST　FD20　RT180"

これは，タートルを画面に表示した後，20前進させ右へ180°(後方向)方向を変えます。

② CMD TURTLE　"HT RP 20[RT90 FD1 LT90 FD1]"

これは，タートルを画面に表示しないで，右90°の方向へ1進んで，左90°の方向へ向きを変えて1進む処理を20回繰り返して図形を描きます。



問題　28　問題27で作成したプログラムは，テンキーだけでタートルを制御するので，細かな図形を描くことができません。そこで，もっと細かな図形が描けるように，設問に従って次のようなキーと動作の処理を追加して下さい。

入力するキー	グラフィックサブコマンド	機　　能
T	ST またはHT	現在，タートルが表示されている場合は消し，消えてる場合は表示します。
F	FDn	タートルを前進させます。前進させる数は，キーボードから入力します。
B	BKn	タートルを後退させます。後退させる数は，キーボードから入力します。
L	LT1	タートルの向きを1°左へ向けます。
R	RT1	タートルの向きを1°右へ向けます。
C	PCn	タートルのペンの色を変更します。変更する色は，キーボードから入力します。
P	PU	現在，タートルが軌跡を描くようになっている場合は描かないように，描かない場合は描くようにします。
M	MVx,y	タートルを移動させます。移動させる位置は，キーボードから入力します。
5	FD1	現在，タートルが向いている方向へタートルを1進めます。

設問 28-1　現在メモリーに入っているプログラムに，次の下線部のような修正を行って下さい。

＜修正と追加＞

```
100 SCREEN 0,0:CMD TURTLE "HT":CMD CLS 3
110 POINT (320,100)
120 CMD TURTLE "ST":TSW=1:PSW=1
130 N$=INPUT$(1):N%=VAL(N$)
132 IF N$>"9" THEN GOSUB *GSC:GOTO 130
140 ON N% GOSUB *N1,*N2,*N3,*N4,*N5,*N6,*N7,*N8,*N9
150 GOTO 130
160 *N1:CMD TURTLE "HD225 FD1":RETURN
```

（つづく）

```
170 *N2:CMD TURTLE "HD180 FD1":RETURN
180 *N3:CMD TURTLE "HD135 FD1":RETURN
190 *N4:CMD TURTLE "HD270 FD1":RETURN
200 *N5:CMD TURTLE "FD1":RETURN
210 *N6:CMD TURTLE "HD90 FD1":RETURN
220 *N7:CMD TURTLE "HD315 FD1":RETURN
230 *N8:CMD TURTLE "HD0 FD1":RETURN
240 *N9:CMD TURTLE "HD45 FD1":RETURN
250 *GSC:' ――― ｸﾞﾗﾌｨｯｸ ｻﾌﾞｺﾏﾝﾄﾞ ―――
260   IF N$><"T" THEN 290
270     IF TSW=1 THEN T$="HT" ELSE T$="ST"
280     CMD TURTLE T$:TSW=1-TSW:RETURN
290   IF N$><"F" THEN 320
300     INPUT "ｽｽﾒﾙｶｽﾞﾊ";N
310     CMD TURTLE "FD(N)":RETURN
320   IF N$><"B" THEN 350
330     INPUT "ｺｳﾀｲｻｾﾙｶｽﾞﾊ";N
340     CMD TURTLE "BK(N)":RETURN
350   IF N$><"L" THEN 370
360     CMD TURTLE "LT1":RETURN
370   IF N$><"R" THEN 390
380     CMD TURTLE "RT1":RETURN
390   IF N$><"C" THEN 420
400     INPUT "ｶﾗｰﾊﾞﾝｺﾞｳ";N
410     CMD TURTLE "PC(N)":RETURN
420   IF N$><"P" THEN 450
430     IF PSW=1 THEN P$="PU" ELSE P$="PD"
440     CMD TURTLE P$:PSW=1-PSW:RETURN
450   IF N$><"M" THEN RETURN
460     INPUT "ｲﾄﾞｳｽﾙｲﾁﾊ    x,y  ";X,Y
470     CMD TURTLE "MV(X),(Y)":RETURN
```

→追加部分 (行番号250～470)

〔修正後の説明〕

① 行番号120の変数TSWは，タートルの表示を行うか否かを決めるスイッチ，変数PSWは軌跡の有無スイッチです。

② 行番号132は，入力した値が数値よりも大きい(英字)ときに，入力されたキーに対する処理を行うようにしています。

③ 行番号200は，数値「5」が入力されたときに，タートルを1進めます。ただし，1ドットしか進まないのでタートルの動く方向は，テンキーの動作と同じ45°単位の方向になります。

④ 行番号250～470は，入力された値にしたがって問題に示されたようなグラフィックサブコマンドを実行します。

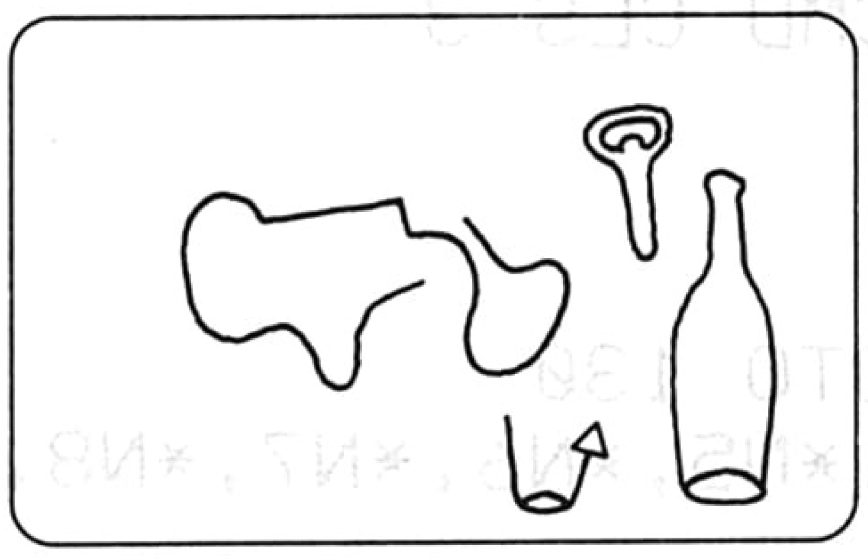
設問 **28-2　プログラムのリストをプリンタに印字して，正しくキーインできているかどうか確認して下さい。**

＜確認用リスト＞

```
100 SCREEN 0,0:CMD TURTLE "HT":CMD CLS 3
110 POINT (320,100)
120 CMD TURTLE "ST":TSW=1:PSW=1
130 N$=INPUT$(1):N%=VAL(N$)
132 IF N$>"9" THEN GOSUB *GSC:GOTO 130
140 ON N% GOSUB *N1,*N2,*N3,*N4,*N5,*N6,*N7,*N8,*N9
150 GOTO 130
160 *N1:CMD TURTLE "HD225 FD1":RETURN
170 *N2:CMD TURTLE "HD180 FD1":RETURN
180 *N3:CMD TURTLE "HD135 FD1":RETURN
190 *N4:CMD TURTLE "HD270 FD1":RETURN
200 *N5:CMD TURTLE "FD1":RETURN
210 *N6:CMD TURTLE "HD90 FD1":RETURN
220 *N7:CMD TURTLE "HD315 FD1":RETURN
230 *N8:CMD TURTLE "HD0 FD1":RETURN
240 *N9:CMD TURTLE "HD45 FD1":RETURN
250 *GSC:' ──── ｸﾞﾗﾌｨｯｸ ｻﾌﾞｺﾏﾝﾄﾞ ────
260  IF N$><"T" THEN 290
270     IF TSW=1 THEN T$="HT" ELSE T$="ST"
280     CMD TURTLE T$:TSW=1-TSW:RETURN
290  IF N$><"F" THEN 320
300     INPUT "ｽｽﾒﾙｶｽﾞﾊ";N
310     CMD TURTLE "FD(N)":RETURN
320  IF N$><"B" THEN 350
330     INPUT "ｺｳﾀｲｻｾﾙｶｽﾞﾊ";N
340     CMD TURTLE "BK(N)":RETURN
350  IF N$><"L" THEN 370
360     CMD TURTLE "LT1":RETURN
370  IF N$><"R" THEN 390
380     CMD TURTLE "RT1":RETURN
390  IF N$><"C" THEN 420
400     INPUT "ｶﾗｰﾊﾞﾝｺﾞｳ";N
410     CMD TURTLE "PC(N)":RETURN
420  IF N$><"P" THEN 450
430     IF PSW=1 THEN P$="PU" ELSE P$="PD"
440     CMD TURTLE P$:PSW=1-PSW:RETURN
450  IF N$><"M" THEN RETURN
460     INPUT "ｲﾄﾞｳｽﾙｲﾁﾊ   x,y ";X,Y
470     CMD TURTLE "MV(X),(Y)":RETURN
```

設問 28-3　正しくキーインできていたら，プログラムを実行していろいろな図形を描いてみて下さい。(入力する英字は大文字にします)

＜実行結果例＞



〔実行後の説明〕

① グラフィックサブコマンドの追加によって，問題27のときよりももっと細かな図形を描くことができます。

② 実行を終了する場合は，STOP キーを押して下さい。

設問 28-4　画面をクリアしてからこのプログラムをドライブ2のフロッピーディスクへ「MONDAI.28」という名前で保存して下さい。

＜実行結果例＞

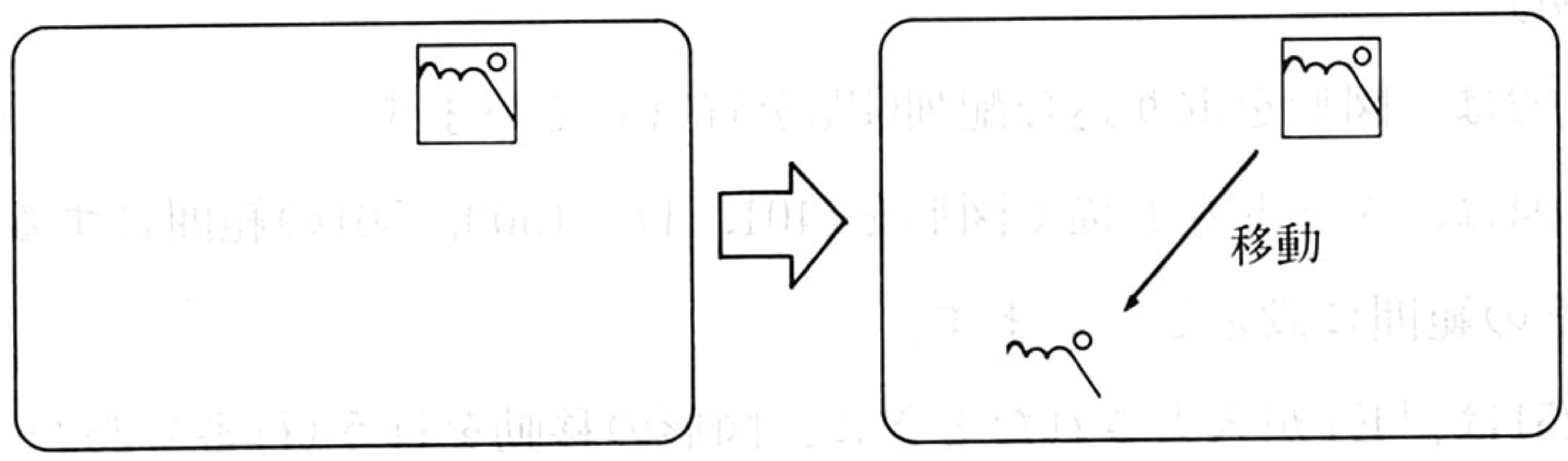
```
CMD CLS 3
                 ←――――ここで画面がクリアされる

Ok
save "2:MONDAI.28"
Ok
```

2.7 図形の移動方法

今まで学習してきた方法では，一度表示した図形と同じものを違う場所に表示しようとする場合，同じ命令で座標指定を変えて実行することになります。ここでは，表示されている図形を一度メモリーへ記憶させてこれを任意の位置に表示し直す方法について学習します。

問題 29 座標(401，1)-(500，50)の範囲に，タートルで任意の図形を描き，それを座標(100，150)-(199，199)の範囲に移動して表示して下さい。



【 要点の説明 】

画面上に表示されている図形を，任意の位置に移動させて表示するには,「GET＠」という命令と「PUT＠」という命令を使います。

GET＠命令は，画面に表示されている図形を配列に取り込む命令で，PUT＠命令は，GET＠命令で配列に取り込んだ図形を画面に表示する命令です。

ここでは，タートルを使って座標(401，1)-(500，50)の範囲に図形を描いて，それをGET＠命令で配列に取り込んで，PUT＠命令で座標(100，150)の位置から表示するようにします。

タートルを使って図形を描くには，問題28で作成したプログラムを利用しますが，図形を描く範囲が(401，1)－(500，50)となるので，この範囲をビューポートに設定してから行います。

図形を描き終わった後は，ビューポートを元に戻して，GET＠命令とPUT＠命令で図形の移動を行います。

GET＠命令で，図形を取り込んでおく配列は，整数型の配列でその大きさは次の計算式で求めます。

　＜配列の大きさ＞＝(((＜横のドット数＞＋7)￥8)＊＜縦のドット数＞＊3＋4)￥2＋1

したがって，ここでの配列の大きさは次のようになります。

　(((100＋7)￥8)＊50＊3＋4)￥2＋1＝978

　　50：縦のドット数

　　100：横のドット数

⇩

DIM　Q%(978)

設問 29-1　メモリーに入っているプログラムに，次の命令を追加して下さい。

＜追加する命令＞

```
102 DIM G%(978)
104 VIEW (401,1)-(500,50),,7
251  IF N$><"E" THEN 260
252     VIEW (0,0)-(639,199)
253     GET@ (401,1)-(500,50),G%
254     PUT@ (100,150),G%
255     END
```

〔追加後の説明〕

① 行番号102は，図形を取り込む配列G%を宣言しています。

② 行番号104は，タートルで描く図形を(401, 1)－(500, 50)の範囲にするために、ビューポートをその範囲に設定しています。

③ 行番号251は、「E」が入力されたときに、図形の移動を行う(行番号252～254)ようにしています。

④ 行番号252は、ビューポートを元に戻しています。

⑤ 行番号253は、座標(401, 1)-(500, 50)の範囲に描れている図形を配列G%に取り込んでいます。このときに取り込まれる図形には，ビューポートの境界線は含まれません。配列は変数名だけを指定して添字は指定しません。

⑥ 行番号254は，配列G%に取り込んだ図形を座標(100, 150)の位置から表示します。

設問 29-2　行番号110を次のように修正して下さい。

```
110 POINT (50,25)
```

〔修正後の説明〕

① タートルを最初に表示する位置が，ビューポート内の中心になるようにLPを変更します。

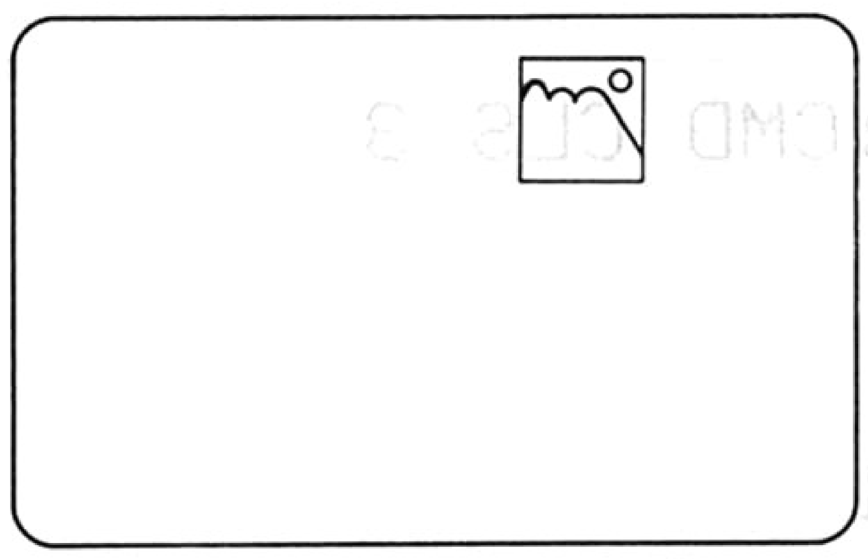
設 問 29-3　プログラムのリストをプリンタに印字して，正しくキーインできているかどうか確認して下さい。

＜確認用リスト＞

```
100 SCREEN 0,0:CMD TURTLE "HT":CMD CLS 3
102 DIM G%(978)
104 VIEW (401,1)-(500,50),,7
110 POINT (50,25)
120 CMD TURTLE "ST":TSW=1:PSW=1
130 N$=INPUT$(1):N%=VAL(N$)
132 IF N$>"9" THEN GOSUB *GSC:GOTO 130
140 ON N% GOSUB *N1,*N2,*N3,*N4,*N5,*N6,*N7,*N8,*N9
150 GOTO 130
160 *N1:CMD TURTLE "HD225 FD1":RETURN
170 *N2:CMD TURTLE "HD180 FD1":RETURN
180 *N3:CMD TURTLE "HD135 FD1":RETURN
190 *N4:CMD TURTLE "HD270 FD1":RETURN
200 *N5:CMD TURTLE "FD1":RETURN
210 *N6:CMD TURTLE "HD90 FD1":RETURN
220 *N7:CMD TURTLE "HD315 FD1":RETURN
230 *N8:CMD TURTLE "HD0 FD1":RETURN
240 *N9:CMD TURTLE "HD45 FD1":RETURN
250 *GSC:' ───── ｸﾞﾗﾌｨｯｸ ｻﾌﾞｺﾏﾝﾄﾞ ─────
251  IF N$><"E" THEN 260
252     VIEW (0,0)-(639,199)
253     GET@ (401,1)-(500,50),G%
254     PUT@ (100,150),G%
255     END
260  IF N$><"T" THEN 290
270     IF TSW=1 THEN T$="HT" ELSE T$="ST"
280     CMD TURTLE T$:TSW=1-TSW:RETURN
290  IF N$><"F" THEN 320
300     INPUT "ｽｽﾒﾙｶｽﾞﾊ";N
310     CMD TURTLE "FD(N)":RETURN
320  IF N$><"B" THEN 350
330     INPUT "ｺｳﾀｲｻｾﾙｶｽﾞﾊ";N
340     CMD TURTLE "BK(N)":RETURN
350  IF N$><"L" THEN 370
360     CMD TURTLE "LT1":RETURN
370  IF N$><"R" THEN 390
380     CMD TURTLE "RT1":RETURN
390  IF N$><"C" THEN 420
400     INPUT "ｶﾗｰﾊﾞﾝｺﾞｳ";N
410     CMD TURTLE "PC(N)":RETURN
420  IF N$><"P" THEN 450
430     IF PSW=1 THEN P$="PU" ELSE P$="PD"
440     CMD TURTLE P$:PSW=1-PSW:RETURN
450  IF N$><"M" THEN RETURN
460     INPUT "ｲﾄﾞｳｽﾙｲﾁﾊ    x,y  ";X,Y
470     CMD TURTLE "MV(X),(Y)":RETURN
```

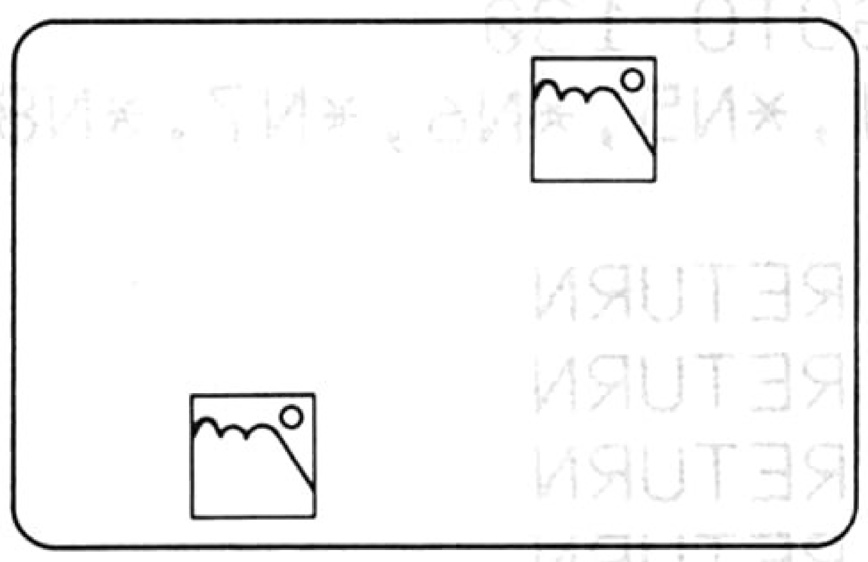
設問 29-4　正しくキーインできていたら，プログラムを実行して下さい。

＜実行結果例＞



・タートルを動かして描いた図形

⇩　・「E」キーを押します。



・タートルで描いた図形と同じ図形が，画面の左下に表示されます。

〔実行後の説明〕

① 一度，PUT命令で図形を表示した後も，配列G%の中の図形は消えません。したがって，これを利用すれば更に他の位置へいくつも同じ図形を表示することができます。

設問 29-5　画面をクリアしてから，このプログラムをドライブ2のフロッピーディスクへ「MONDAI.29」という名前で保存して下さい。

＜実行結果例＞

```
CMD CLS 3
                              ←――――― ここで画面がクリアされる

Ok
save "2:MONDAI.29"
Ok
```

【 命令のまとめ 】

① GET＠の一般形式と書き方例を次に示します。

＜一般形式＞

GET＠(X_1, Y_1)-(X_2, Y_2), 配列変数名

・座標(X_1, Y_1)-(X_2, Y_2)の範囲内に描れている図形を, 指定された配列変数に取り込みます。

・指定する座標は, ビューポート内のスクリーン座標でなければなりません。

・配列変数名は, 数値型(整数型, 単精度型, 倍精度型のいずれか)のものを指定します。

・指定する配列の大きさは, 取り込もうとする図形の範囲によって変わります。その大きさは, 次の計算式で求めます。

(((＜横のドット数＞+7)￥8)＊＜縦のドット数＞＊N+4)￥M+1

・Mは配列変数の型によって変わります。

整数型	M＝2
単精度型	M＝4
倍精度型	M＝8

・Nは画面モードによって変わります。

白黒モード	N＝1
カラーモード	N＝3

＜書き方例＞

① GET＠(200, 50)－(300, 120), A

これは, 座標(200, 50)-(300, 120)の範囲内に描かれている図形を, 単精度の配列Aに取り込みます。

② GET＠(340, 120)-STEP(50, 20), G%

これは, 座標(340, 120)からX方向に50, Y方向に20の範囲内に描かれている図形を, 整数型の配列G%に取り込みます。

② PUT@命令の一般形式と書き方例を次の示します。

<一般形式>

PUT@(X, Y), 配列変数名, 条件, フォアグラウンドカラー, バックグラウンドカラー

・座標(X, Y)の位置に, 指定された配列変数名の中に取り込まれている図形を表示します。

・指定する座標は, GET@命令と同じようにビューポート内のスクリーン座標でなければなりません。

・配列変数名は, GET@命令で図形を取り込んだ配列の名前を指定します。PUT@命令では, GET@命令と違って, 配列変数名に添字を指定することができます。
添字を指定した場合, 表示される図形は取り込んだ図形の途中からになります。

・条件は, 図形を表示するときに, いろいろな機能を付けるもので, 次の種類があります。

PSET	配列内の図形をそのまま表示します。
PRESET	図形を表示する位置に, 既に何か描かれている場合, それを消してから表示を行います。 PRESETを指定すると, フォアグラウンドカラーとバックグラウンドカラーは「7−(指定した色の番号)」の色で表示されます。
OR AND XOR	表示しようとする位置に既に表示されている図形との論理演算を行い, その結果を表示します。

・条件を省略した場合は, 「XOR」の指定になります。

・フォアグラウンドカラーとバックグラウンドカラーは, 白黒モードで配列に取り込んだ図形をカラーモードで表示するときに, 図形に付ける色を指定します。

<書き方例>

① PUT@(400, 60), A

これは, GET@命令で配列変数Aに取り込んだ図形を, 座標(400, 60)の位置から表示します。

② PUT@(500, 120), G%(50)

これは, GET@命令で配列変数G%に取り込んだ図形のうち添字50以降の図形を, 座標(500, 120)の位置に表示します。

2.8 模様の付け方

N$_{88}$DISK-BASICでは，点数や一点鎖線といった規則的な模様の繰り返しで線を表示したり，図形の内側をいろいろな模様で塗りつぶしたりすることができます。

これらは，既に学習してきた「LINE命令」や「PRINT命令」で表示する模様を指定することによって行います。

LINE命令では，いろいろな模様の線を表示することができます。この場合の模様の付いた線のことを「ラインスタイル」といいます。

PAINT命令では，設定した模様(これを「タイルパターン」と呼びます)で図形の内側を塗りつぶします。このように，タイルパターンで図形を塗りつぶしていくことを「タイリング」と呼びます。

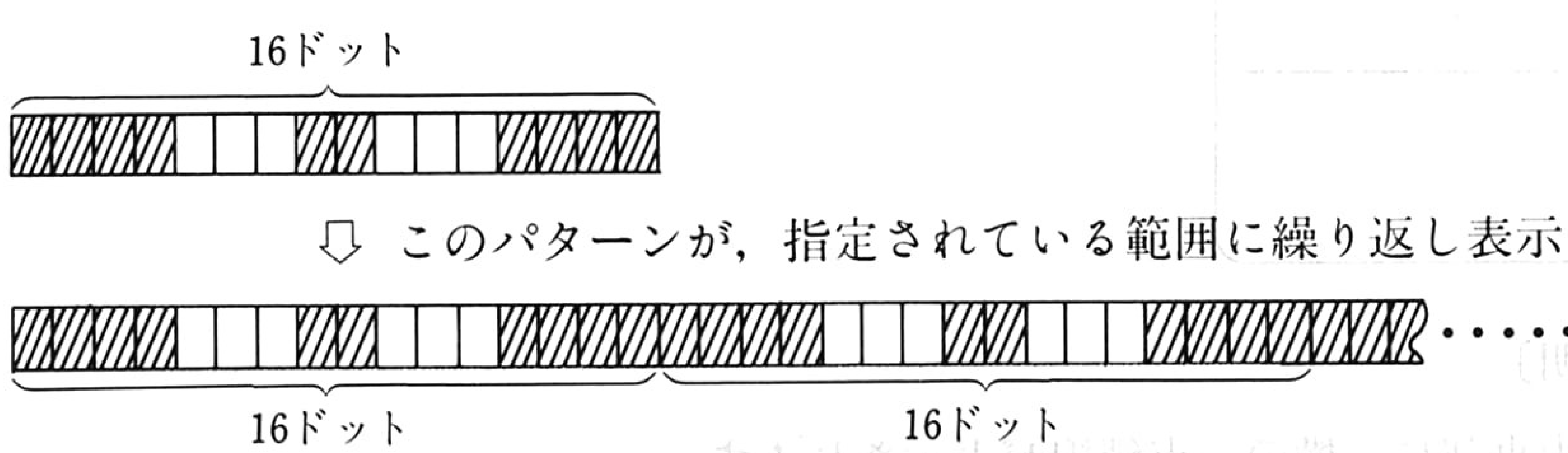
2.8.1 ラインスタイルの設定

問題 30 座標(0, 99)－(639, 99)に，次のような一点鎖線を表示して下さい。

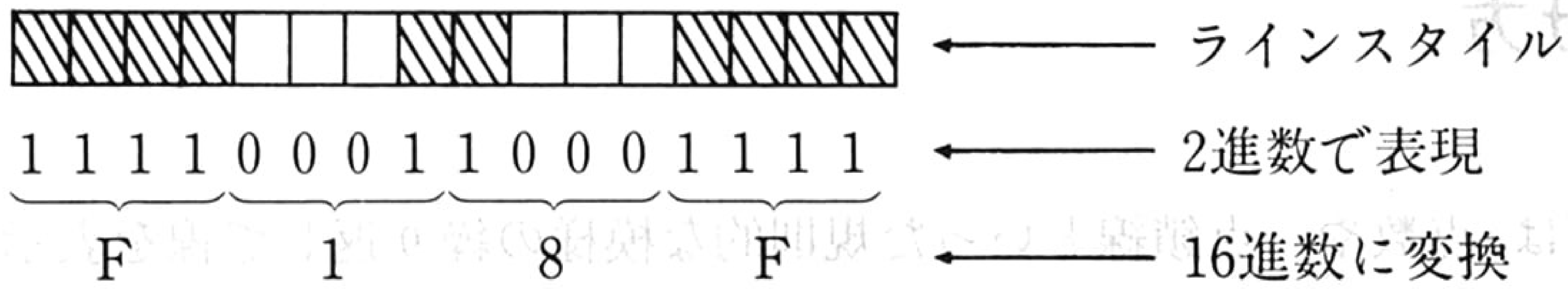
【 要点の説明 】

模様線の表示は，普通の直線を表示するときと同じようにLINE命令で行いますが，この場合，表示する線の模様(ラインスタイル)を16進数で指定します。

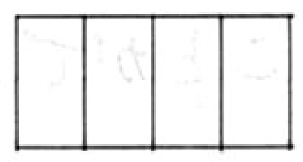
ラインスタイルは，次のように16ドット分を1パターンとするもので，LINE命令を実行すると，この1パターン(16ドット)が繰り返されて表示されます。



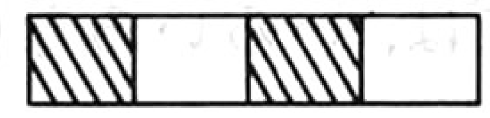
上の図は，問題の一点鎖線を表示する場合のラインスタイルを16個のマス目で表したものです。ところが，LINE命令では，これを16進数で指定しなければなりません。そこで，一度ラインスタイルの各ドットを2進数で表現(表示するドットを1，表示しないドットを0とする)して，それをさらに16進数に変換します。



※ 2進数を16進数に変換する場合は，2進数の4ビットごとに次のような重み付けを行って10進数にした後，対応する16進数にします。



8 4 2 1 ← 重み

(例) 

⇩

1×8+0×4+1×2+0×1 重み付け
=8+0+2+0
=10

⇩

0A(16)

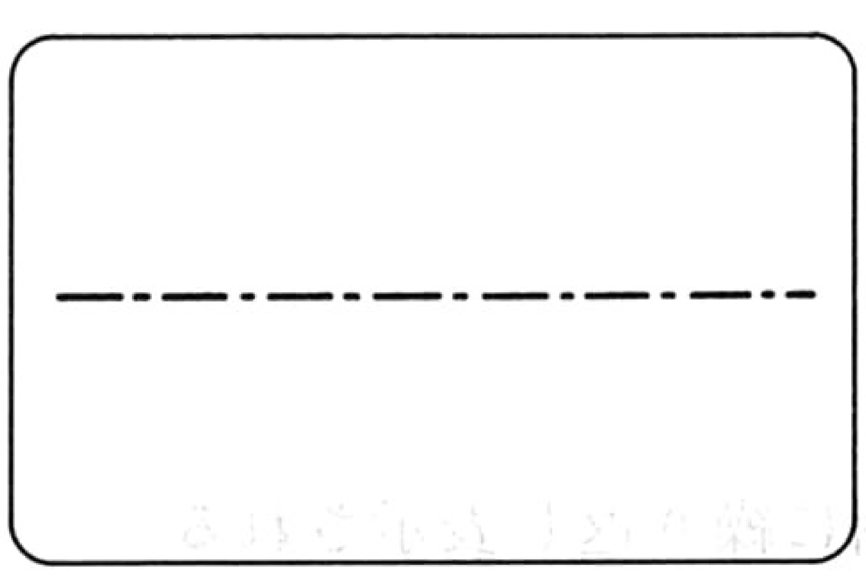
これで，一点鎖線を表示する場合のラインスタイルの16進表現ができました。あとは，この16進数(F18F)の頭に「＄H」を付けて，LINE命令で指定します。

設問 30-1 画面をクリアしてから次の命令を実行して，問題の一点鎖線を表示して下さい。

<実行する命令>

```
LINE (0,99)-(639,99),,,&HF18F
```

<実行結果例>



〔実行後の説明〕

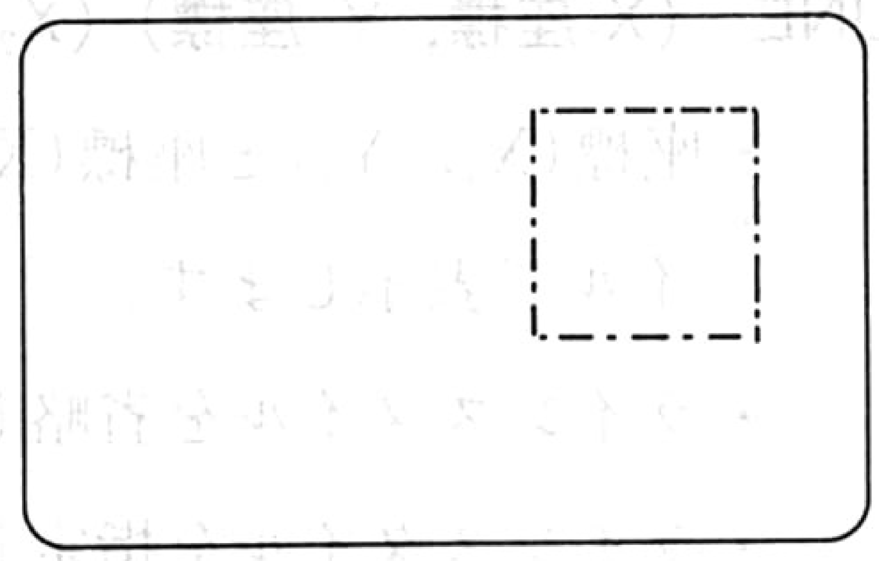
① 画面の中央部に，横の一点鎖線が表示されます。

設問 30-2 画面をクリアした後，座標(400, 30)-(600, 130)を対角線の頂点とする四角形を一点鎖線で表示して下さい。

〔操作説明〕

・命令は，「LINE (400, 30)-(600, 130), , B, &HF18F」をキーインします。

＜実行結果例＞

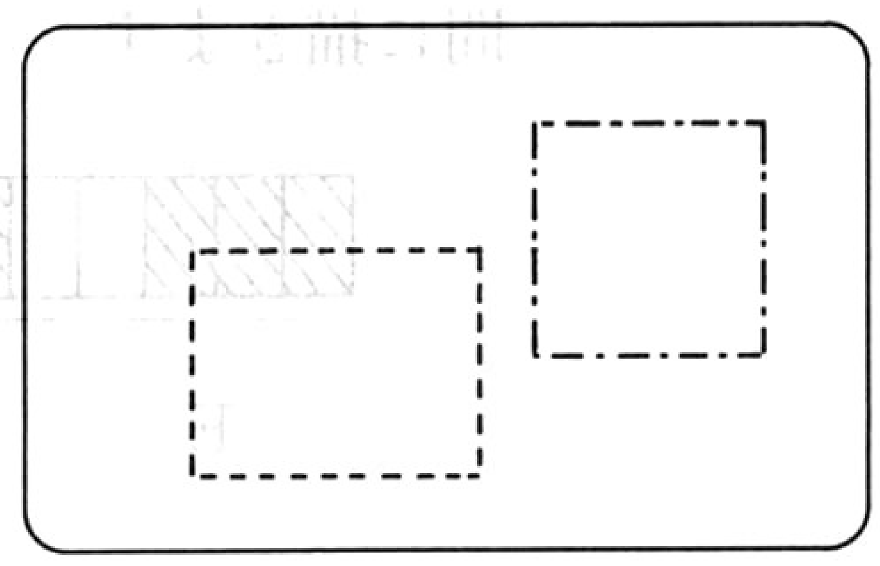


設問 30-3 座標(100, 80)と(350, 180)を対角線の頂点とする四角形を点線で表示して下さい。

〔操作説明〕

・命令は，「LINE (100, 80)-(350, 180), , B, &HF0F0」をキーインします。

＜実行結果例＞



〔実行後の説明〕

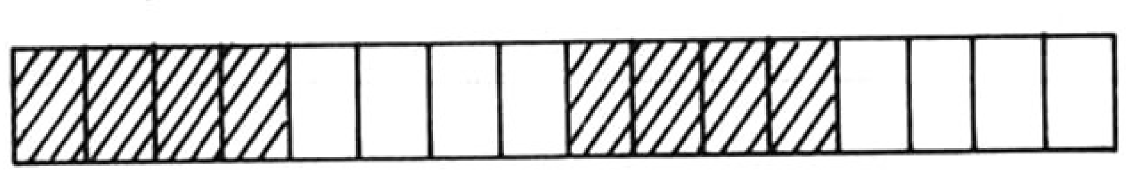
① ここで指定した点線のラインスタイル(&HF0F0)は，次のような模様です。

F　0　F　0

↓　↓　↓　↓

1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0

⇩　⇩　⇩　⇩



【 命令のまとめ 】

いろいろな模様の線を描くLINE命令の一般形式と書き方例を次に示します。

<一般形式>

LINE (X1座標, Y1座標)-(X2座標, Y2座標), パレット番号, B, ラインスタイル

・座標(X1, Y1)と座標(X2, Y2)の位置を結ぶ線または四角形を，指定したラインスタイルで表示します。

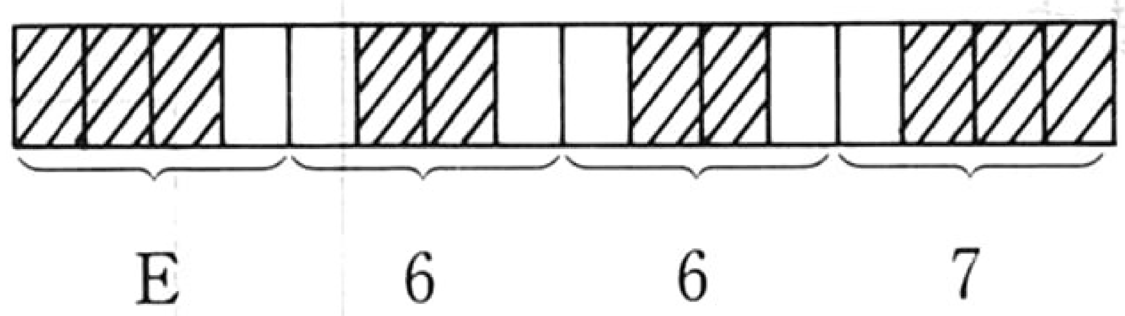
・ラインスタイルを省略したときは，通常のLINE命令と同じ使い方になります。

・ラインスタイルを指定したときは，「BF」の指定はできません。

<書き方例>

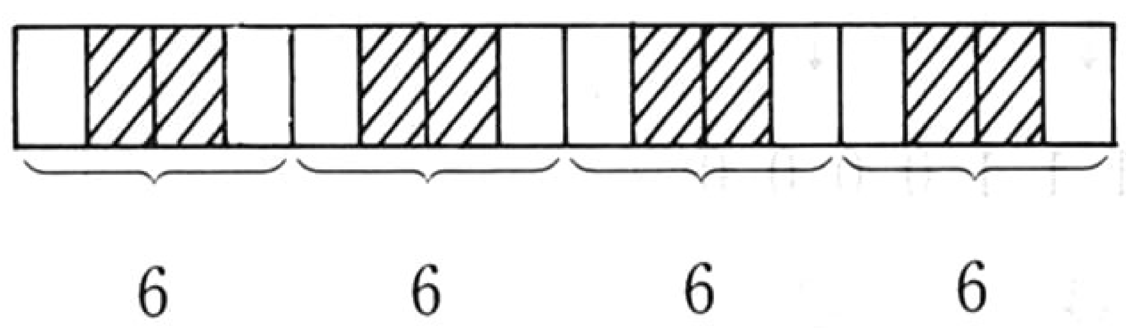
① LINE (300, 20)-(500, 20), 2, , &HE667

これは，次の図のような二点鎖線を赤色で座標(300, 20)と座標(500, 20)の間に描きます。



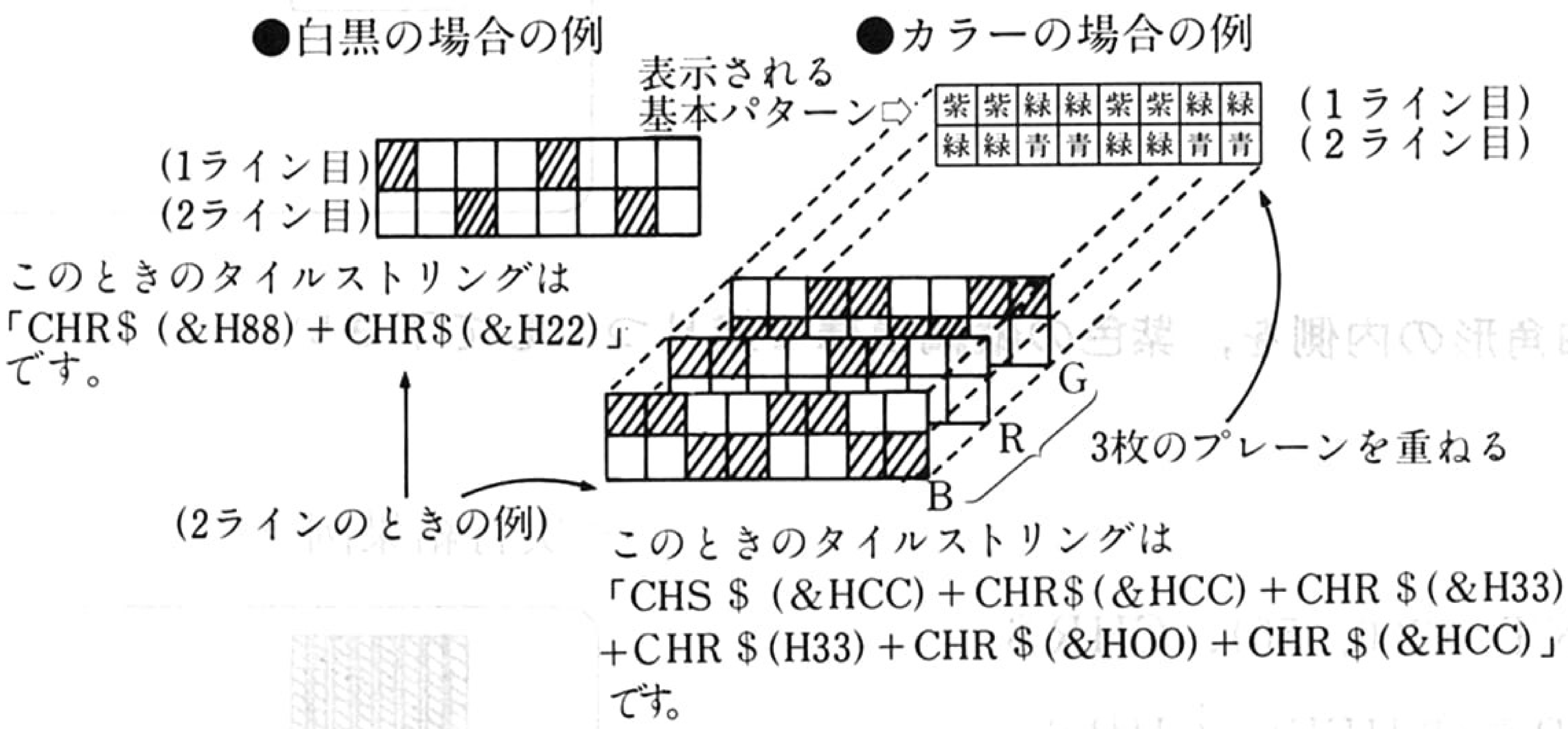
② LINE (300, 30)-(500, 100), 4, B, &H6666

これは，座標(300, 30)と座標(500, 100)を対角線の頂点とする四角形を，次の図のような緑色の点線で描きます。



2.8.2 タイリング

「タイリング」は，表示した図形の内側を更に特殊な模様で塗ることであり，このときに指定する基本模様を「タイルストリング」といいます。タイルストリングは，次の図のように白黒の場合「8ドット×ライン数」，カラーの場合「8ドット×3プレーン×ライン数」で模様の基本パターンを決めます。

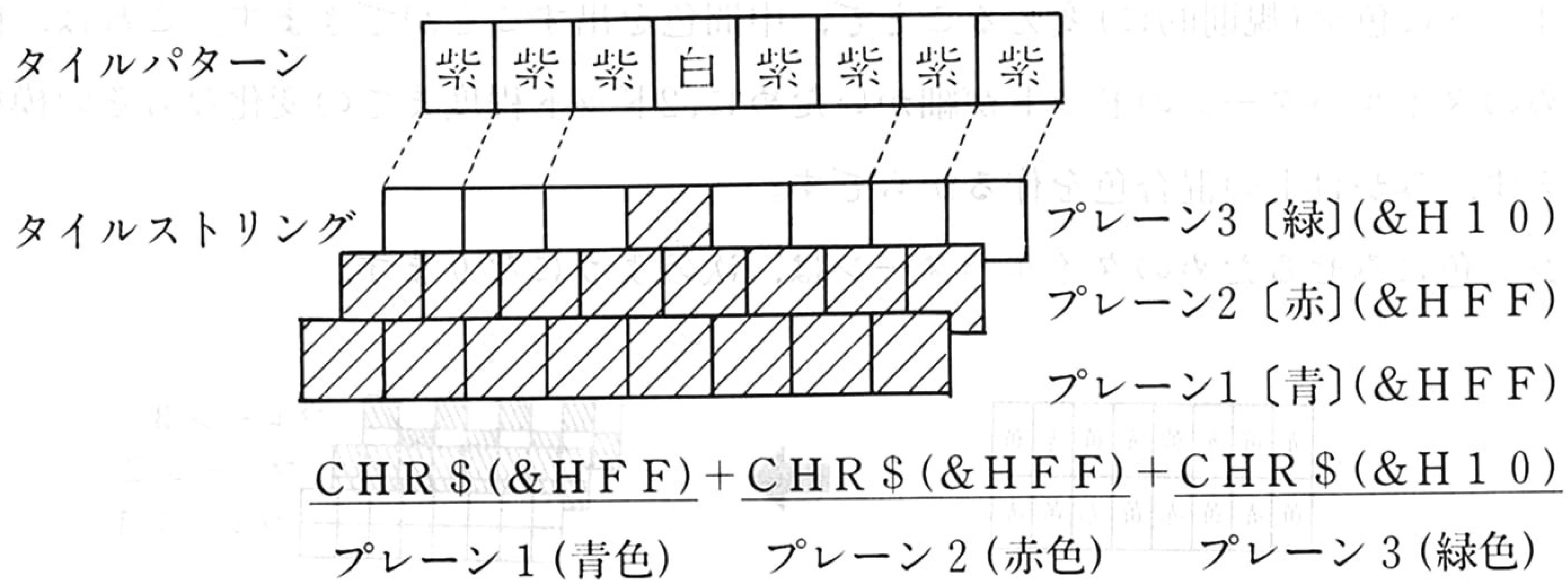


タイリングは，「PRINT命令」でタイルストリングを指定することにより行います。タイリングを行うことにより図形にいろいろな模様を付けることはもちろん，カラーモードでは，いろいろなドットの組み合わせで混合色(中間色)を作り出すことができます。

> **問題 31** 座標(100, 0)と座標(300, 100)を対角線の頂点とする四角形を描いた後，その中を紫色の縦縞模様で塗りつぶして下さい。

【説　明】

紫色の縦縞模様となるようなタイルストリングを考えます。ここでは，白地に紫色の太い縦縞を付けることにします。

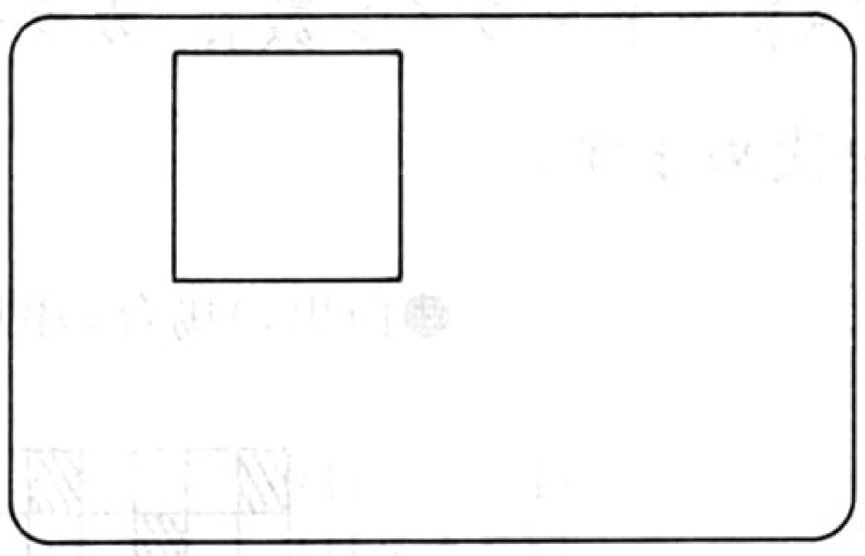


設問 31-1　画面をクリアしてから，問題13の四角形を描いて下さい。

〔操作説明〕

・命令は，「LINE　(100，0)-(300，100)，，B」をキーインします。

＜実行結果例＞

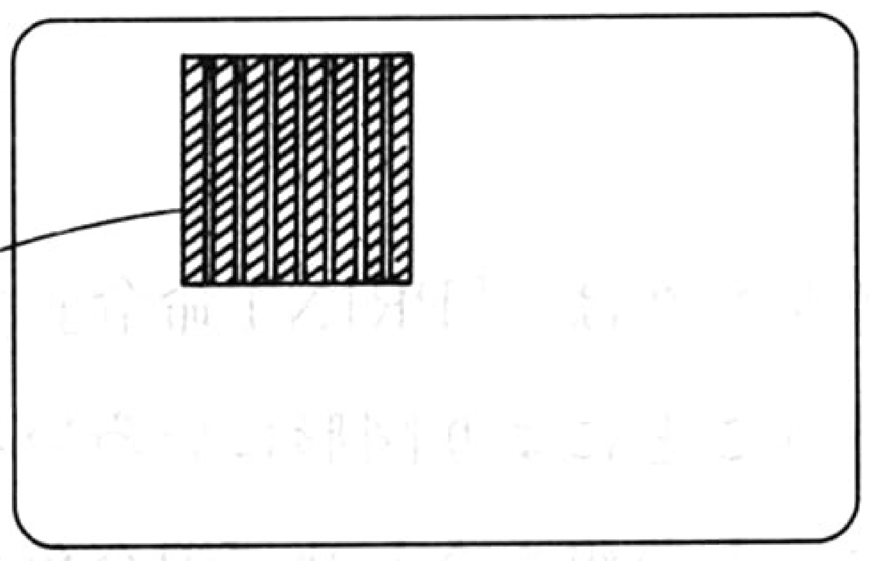


設問 31-2　描いた四角形の内側を，紫色の縦縞模様で塗りつぶして下さい。

〔操作説明〕

・命令は，「PAINT　(200，50)，CHR＄(&HFF)＋CHR＄(&HFF)＋CHR＄(&H10)，7」をキーインします。

＜実行結果例＞



〔実行後の説明〕

① 実行例では，タイルストリングをPAINT命令で直接指定しましたが，文字列にタイルストリングを入れておいて実行しても構いません。

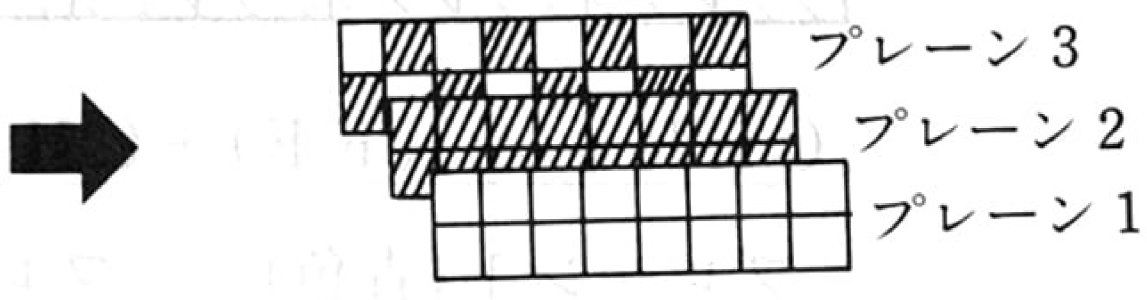
> **問題　32**　座標(500，100)を中心とする半径100の黄色の円を描いた後，その内側をオレンジ色で塗りつぶして下さい。

【説　明】

ドットごとに色を(規則的に)変えることで，中間色を出すことができます。これは，模様を付けるためのタイルパターンのドットが細かいために，2ドット程度までの変化ならその模様がはっきり見えず，みかけ上の混合色を作るからです。

オレンジ色にみせるためのタイルパターンは，次のようになります。

赤	黄	赤	黄	赤	黄	赤	黄
黄	赤	黄	赤	黄	赤	黄	赤



設問 32-1 オレンジ色のタイルパターンのタイルストリングを，変数TILE＄に入れて下さい。

＜実行結果例＞

```
TILE$=CHR$(&H0)+CHR$(&HFF)+CHR$(&H55)
```

〔実行後の説明〕

① 必ずプレーン1(青)，プレーン2(赤)，プレーン3(緑)の順に指定します。

② 表示をしないプレーンがあっても，そのプレーンのタイルストリングの指定を省略することはできません。

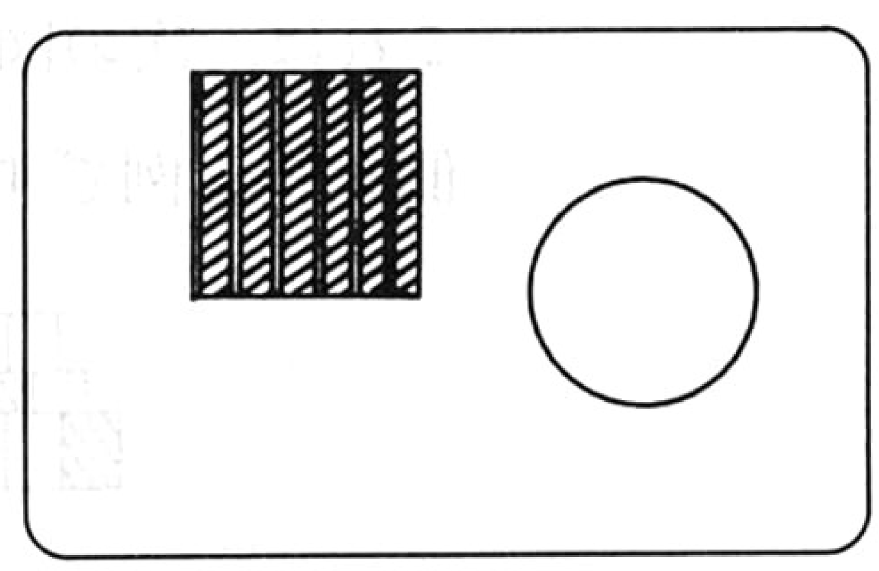
③ オレンジ色に見せるためには，赤色と黄色のドットを交互に表示すればよいわけですから，赤色と黄色のドットが交互になるように2行分のタイルストリングを指定します。

設問 32-2 座標(500，100)を中心とする半径100の黄色の円を描いて下さい。

〔操作説明〕

・命令は，「CIRCLE (500，100)，100，6」をキーインします。

＜実行結果例＞

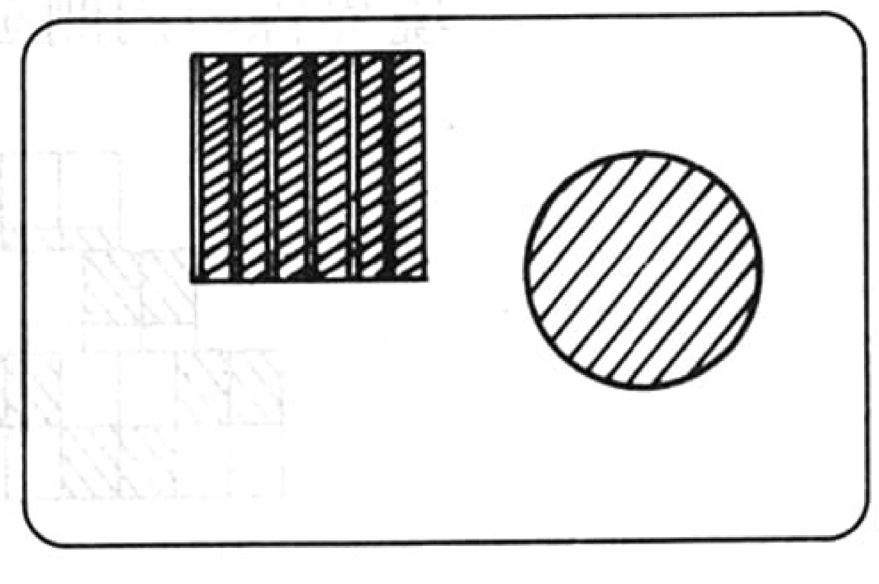


設問 32-3 円の内側をオレンジ色で塗りつぶして下さい。

〔操作説明〕

・命令は，「PAINT (500，100)，TILE＄，6」をキーインします。

＜実行結果例＞



【 命令のまとめ 】

図形の内側を特殊な模様で塗りつぶすPAINT命令の一般形式と書き方例を次に示します。

＜一般形式＞

PAINT （X座標，Y座標），タイルストリング，境界色

・座標(X，Y)の位置を中心に，指定した境界色で表示されている図形の内側をタイルストリングで塗ります。

・タイルストリングは，表示する模様の基本パターンを指定します。

・基本パターンの指定は，8ドットのパターンを1ラインとして行います。

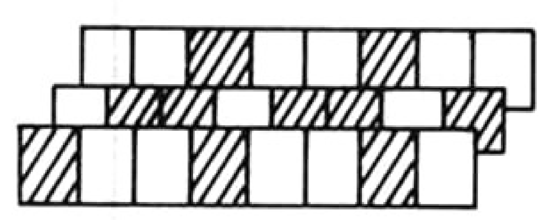
・基本パターンを複数ライン分指定した場合は，その複数ラインで一つのパターンになります。

・カラーモードの場合，プレーン1からプレーン3まで順にドットパターンを指定して1ラインになります。

＜書き方例＞

①
```
A$=CHR$(&H92)+CHR$(&H6D)+CHR$(&H24)
LINE (100, 50)-(300, 150), 4, B
PAINT (150, 100), A$, 4
```

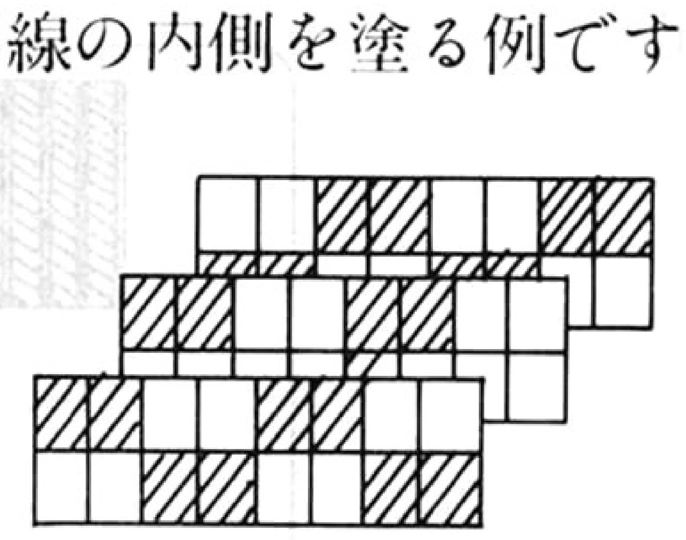
これは，次の図のようなドット構成のタイルストリングで緑色の境界線の内側を塗る例です。



プレーン3〔緑〕(&H24)
プレーン2〔赤〕(&H6D)
プレーン1〔青〕(&H92)

②
```
B$=CHR$(&HCC)+CHR$(&HCC)+CHR$(&H33)
B$=B$+CHR$(&H33)+CHR$(&H0)+CHR$(&HCC)
CIRCLE (400, 100), 80, 7
PAINT (400, 100), B$, 7
```

これは，次の図のようなドット構成の2ライン分のタイルストリングで白色の境界線の内側を塗る例です。



プレーン3〔緑〕1ライン目(&H33)
プレーン3〔緑〕2ライン目(&HCC)
プレーン2〔赤〕1ライン目(&HCC)
プレーン2〔赤〕2ライン目(&H0)
プレーン1〔青〕1ライン目(&HCC)
プレーン1〔青〕2ライン目(&H33)

＊ 2ライン以上のときは，必ずラインごとに青，赤，緑の順に指定します。

問題　33　問題29で作成した，タートルで図形を描くプログラムに，ペイント機能(タイリングを含む)を追加して下さい。

ペイントを行うときの指示キーは，「@(アットマーク)」にします。

【 要点の説明 】

問題29のプログラムは，現在メモリーに入っています。

ペイントの指示キー「@」が入力されたら，領域色と境界色を入力してペイントを行うようにします。領域色として，「0」～「7」以外の数値が入力されたらタイリングを行うものとして，ライン数とタイルストリングを入力するようにします。

設問 33-1　画面をクリアしてから，次のような修正と追加を行って下さい。

＜修正＞

```
130 CLS:N$=INPUT$(1):N%=VAL(N$)
450  IF N$><"M" THEN 480
```

＜追加＞

```
480   IF N$><"@" THEN RETURN
490      INPUT "ｷｮｳｶｲｾﾝ ﾉ ｲﾛ";C2
500      IF C2<0 OR C2>7 THEN 490
510      INPUT "ﾇﾘﾂﾌﾞｽ ｲﾛ";C1
520      IF C1<0 OR C1>7 THEN 540
530         PAINT STEP(0,0),C1,C2:RETURN
540      INPUT "ﾀｲﾙﾊﾟﾀｰﾝ ﾉ ﾗｲﾝｽｳ";L
550      TILE$=""
560      FOR I=1 TO L
570         INPUT "ﾌﾟﾚｰﾝ1 ﾉ ﾊﾟﾀｰﾝ (xx) ";A$:A$=CHR$(VAL("&H"+A$))
580         INPUT "ﾌﾟﾚｰﾝ2 ﾉ ﾊﾟﾀｰﾝ (xx) ";B$:B$=CHR$(VAL("&H"+B$))
590         INPUT "ﾌﾟﾚｰﾝ3 ﾉ ﾊﾟﾀｰﾝ (xx) ";C$:C$=CHR$(VAL("&H"+C$))
600         TILE$=TILE$+A$+B$+C$
610      NEXT I
620      PAINT STEP(0,0),TILE$,C2:RETURN
```

〔修正および追加後の説明〕

① 画面に表示されている文字をクリアするために，行番号130で指示キーを入力する前にテキスト画面をクリアするようにします。

② 行番号480は，「@」が入力されたときにペイントを行うようにします。

③ 行番号490と500は，ペイントを行うときの境界色を入力します。

④ 行番号510と520は，塗りつぶす色(領域色)を入力します。入力が「0」～「7」の範囲であれば，行番号530でペイントを行いますが，それ以外の場合は，行番号540以下でタイリングを行うようにします。

⑤ 行番号540は，タイルパターンのライン数を入力します。

⑥ 行番号550は，タイルストリングを入れておくための変数TILE＄をクリアします。

⑦ 行番号560～610は，指定されたライン数だけのタイルストリングを入力します。入力は，ドットパターンの16進コードを文字列として入力した後，VAL関数で10進数に変換しそれを対応する文字列に変換します。

⑧ 行番号620は，入力したタイルストリングでタイリングを行います。

設問 33-2 プログラムのリストをプリンタに印字して，正しくキーインできているかどうか確認して下さい。

＜確認用リスト＞

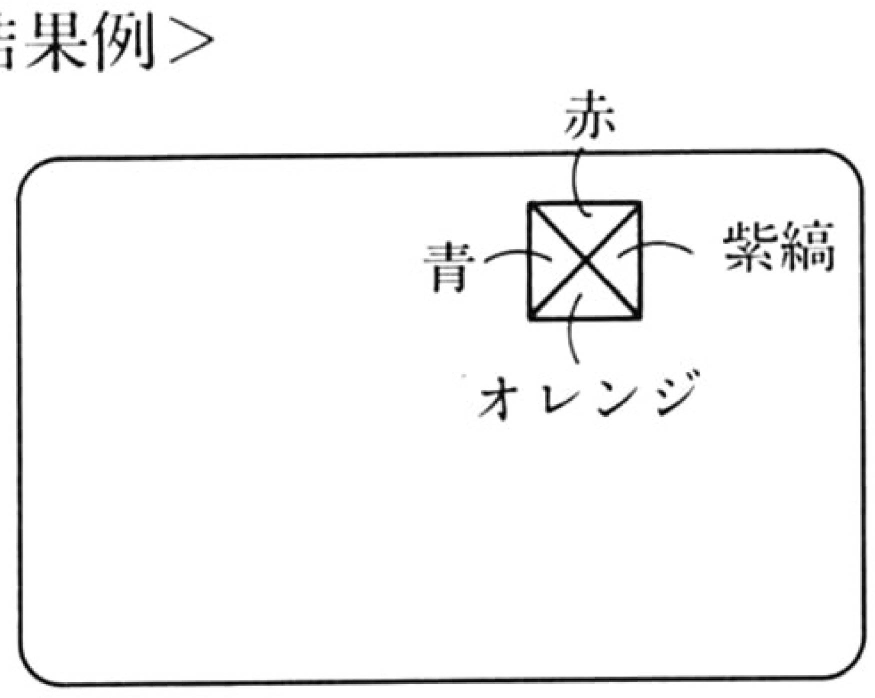
```
100 SCREEN 0,0:CMD TURTLE "HT":CMD CLS 3
102 DIM G%(978)
104 VIEW (401,1)-(500,50),,7
110 POINT (50,25)
120 CMD TURTLE "ST":TSW=1:PSW=1
130 CLS:N$=INPUT$(1):N%=VAL(N$)
132 IF N$>"9" THEN GOSUB *GSC:GOTO 130
140 ON N% GOSUB *N1,*N2,*N3,*N4,*N5,*N6,*N7,*N8,*N9
150 GOTO 130
160 *N1:CMD TURTLE "HD225 FD1":RETURN
170 *N2:CMD TURTLE "HD180 FD1":RETURN
180 *N3:CMD TURTLE "HD135 FD1":RETURN
190 *N4:CMD TURTLE "HD270 FD1":RETURN
200 *N5:CMD TURTLE "FD1":RETURN
210 *N6:CMD TURTLE "HD90 FD1":RETURN
220 *N7:CMD TURTLE "HD315 FD1":RETURN
230 *N8:CMD TURTLE "HD0 FD1":RETURN
240 *N9:CMD TURTLE "HD45 FD1":RETURN
250 *GSC:'———— ｸﾞﾗﾌｨｯｸ ｻﾌﾞｺﾏﾝﾄﾞ ————
251  IF N$><"E" THEN 260
252     VIEW (0,0)-(639,199)
253     GET@ (401,1)-(500,50),G%
254     PUT@ (100,150),G%
255     END
260  IF N$><"T" THEN 290
270     IF TSW=1 THEN T$="HT" ELSE T$="ST"
280     CMD TURTLE T$:TSW=1-TSW:RETURN
290  IF N$><"F" THEN 320
300     INPUT "ｽｽﾒﾙｶｽﾞﾊ";N
310     CMD TURTLE "FD(N)":RETURN
320  IF N$><"B" THEN 350
330     INPUT "ｺｳﾀｲｻｾﾙｶｽﾞﾊ";N
340     CMD TURTLE "BK(N)":RETURN
```

(つづく)

```
350  IF N$><"L" THEN 370
360     CMD TURTLE "LT1":RETURN
370  IF N$><"R" THEN 390
380     CMD TURTLE "RT1":RETURN
390  IF N$><"C" THEN 420
400     INPUT "ｶﾗｰﾊﾞﾝｺﾞｳ";N
410     CMD TURTLE "PC(N)":RETURN
420  IF N$><"P" THEN 450
430     IF PSW=1 THEN P$="PU" ELSE P$="PD"
440     CMD TURTLE P$:PSW=1-PSW:RETURN
450  IF N$><"M" THEN 480
460     INPUT "ｲﾄﾞｳｽﾙｲﾁﾊ    x,y  ";X,Y
470     CMD TURTLE "MV(X),(Y)":RETURN
480  IF N$><"@" THEN RETURN
490     INPUT "ｷｮｳｶｲｾﾝ ﾉ ｲﾛ";C2
500     IF C2<0 OR C2>7 THEN 490
510     INPUT "ﾇﾘﾂﾌﾞｽ ｲﾛ";C1
520     IF C1<0 OR C1>7 THEN 540
530        PAINT STEP(0,0),C1,C2:RETURN
540     INPUT "ﾀｲﾙﾊﾟﾀｰﾝ ﾉ ﾗｲﾝｽｳ";L
550     TILE$=""
560     FOR I=1 TO L
570       INPUT "ﾌﾟﾚｰﾝ1 ﾉ ﾊﾟﾀｰﾝ (xx) ";A$:A$=CHR$(VAL("&H"+A$))
580       INPUT "ﾌﾟﾚｰﾝ2 ﾉ ﾊﾟﾀｰﾝ (xx) ";B$:B$=CHR$(VAL("&H"+B$))
590       INPUT "ﾌﾟﾚｰﾝ3 ﾉ ﾊﾟﾀｰﾝ (xx) ";C$:C$=CHR$(VAL("&H"+C$))
600       TILE$=TILE$+A$+B$+C$
610     NEXT I
620     PAINT STEP(0,0),TILE$,C2:RETURN
```

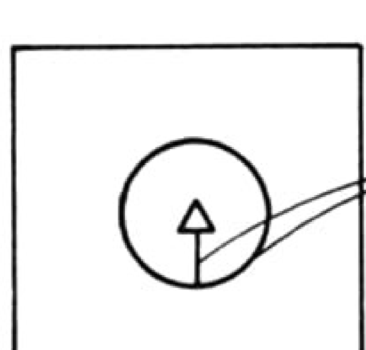
設問 33-3　正しくキーインできていたら，プログラムを実行していろいろな図形を描いて下さい。

＜実行結果例＞



〔実行後の説明〕

① ペイントを行う場合，境界色として指定した色の線上で行わないように注意します。特に，タートルの軌跡と同じ色を境界色とした場合は，軌跡を描きながらペイントを行うことはできません。



タートルが描いた軌跡が白色のときに，境界色を白色としてペイントを行う場合，図のように，タートルが軌跡を描いている時点でのペイントはできません。
この場合は，「P」を入力してタートルのペンを上げた後タートルを境界色の線からずらして行います。

② ペイントを行う場合は，必ずタートルを消してから行って下さい。

設問 33-4　このプログラムをドライブ2のフロッピーディスクへ「MONDAI.33」という名前で保存して下さい。

＜実行結果例＞

```
save "2:MONDAI.33"
Ok
```

2.9 グラフィック図形のセーブとロード

グラフィック画面に描いた図形を，配列変数に記憶させておいて他の場所に移動させる方法は，「2.7 図形の移動方法」で学習しました。これを使うと，GET＠命令で配列に記憶した図形は，プログラムの実行を中断したり終了したりしない限り，画面をクリアしても何回でも再生することができます。

ここでは，配列に図形を記憶できるということを利用して，グラフィック画面に描いた図形をフロッピーディスクにセーブしておき，いつでも同じ図形を画面に表示できるようにする方法を学習します。

問題 34 問題33のプログラムを使って図形を描いた後，その図形をフロッピーディスクにセーブして下さい

【 要点の説明 】

問題33のプログラムは，「E」を入力すると図形を配列に取り込んだあとすぐに違う位置への表示を行い終了するようになっています。そこで，この部分の図形を配列に取り込んだらその配列の値をフロッピーディスクへセーブするように修正します。

配列の値をフロッピーディスクへセーブするには，二つの方法があります。一つは，シーケンシャルファイル(またはランダムファイル)に，配列の値をデータとして書き込んでいく方法です。もう一つの方法は，配列変数の中の値をマシン語とみなして「BSAVE命令」でセーブするという方法です。ここでは，BSAVE命令を使ってセーブする方法を学習します。これによって，データの書き込みや読み込みといった処理をプログラムで行う必要がなくなります。

BSAVE命令は，本来マシン語のプログラムをフロッピーディスクにセーブする命令です。この命令を使ってセーブを行う場合は，セーブしたときの「ファイル名」ならびにセーブするマシン語プログラムのメモリー上の開始番地と長さを指定します。ここでは，配列の中の値をマシン語とみなしてセーブするわけですから，配列の中の値のメモリー上の開始番地とその長さを調べなければなりません。

配列変数がメモリー上のどの位置から格納されているかは，「VARPTR関数」で調べることができます。また，配列変数の長さ(バイト数)はその型と要素の数から求めることができます。

＜配列変数の開始位置の求め方＞

VARPTR(配列変数名(0))

└── OPTION BASEが「1」の場合は「1」を指定する

<配列変数の長さ(バイト数)の求め方>

整数型配列の場合　2＊(添字の最大値＋1)

単精度型配列の場合　4＊(添字の最大値＋1)

倍精度型配列の場合　8＊(添字の最大値＋1)

設問 34-1　行番号254を次のように修正して下さい。

<修正例>

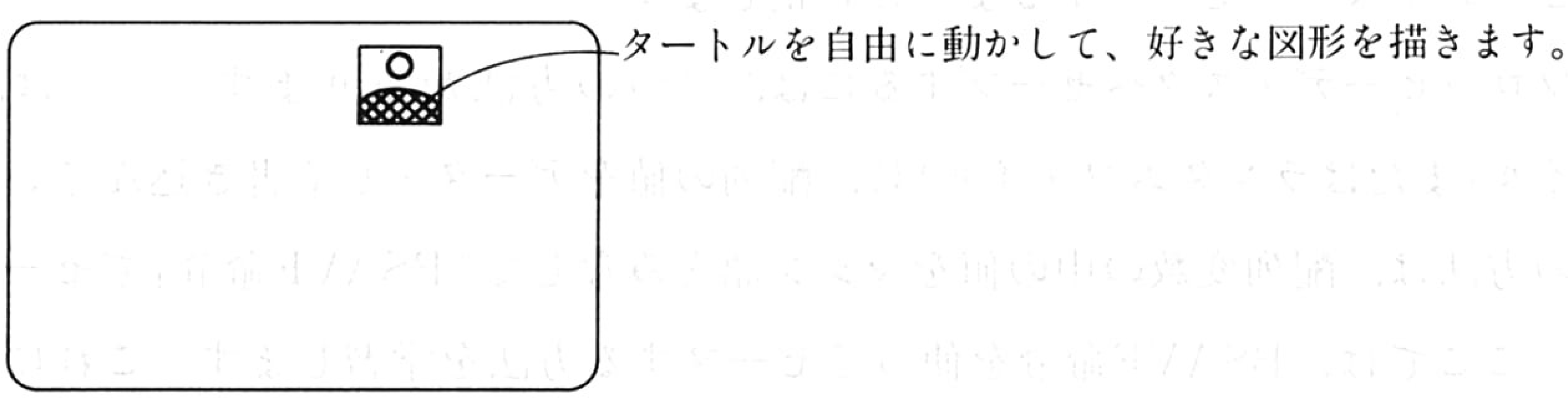
```
254      BSAVE "2:GAMEN.1",VARPTR(G%(0)),2*(978+1)
```

〔修正後の説明〕

① 座標(401, 1)～(500, 50)の範囲に描かれた図形がドライブ2のフロッピーディスクに「GAMEN.1」という名前で，セーブされるようになります。

設問 34-2　正しく修正できたらプログラムを実行して，好きな図形を描き，図形のセーブを行って下さい。

<実行結果例>



〔実行語の説明〕

① 「E」を入力すると，ビューポート内に描いた図形をフロッピーディスクにセーブして実行を終了します。

設問 34-3　このプログラムをドライブ2のフロッピーディスクへ「MONDAI.34」という名前で保存して下さい。

<実行結果例>

```
save "2:MONDAI.34"
Ok
```

設問 34-4　メモリーと画面をクリアしてから，次のプログラムをキーインして下さい。

＜キーインするプログラム＞

```
100 DIM X%(978)
110 SCREEN 0,0:CMD CLS 3
120 BLOAD "2:GAMEN.1",VARPTR(X%(0))
130 PUT@ (300,100),X%
140 END
```

〔キーイン後の説明〕

① 行番号100は，フロッピーディスクにセーブされている図形をメモリーにロードしたときに，その図形を入れるための配列の定義を行っています。配列はセーブしたときと同じ大きさのものを定義します。また，配列名もセーブしたときと同じ型で定義します。

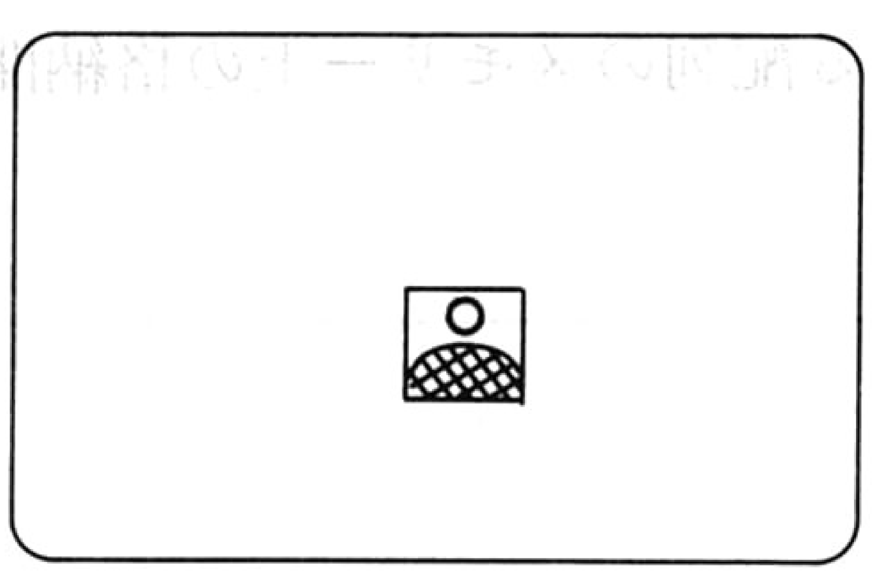
② 行番号120は，ドライブ2のフロッピーデイスクに「GAMEN.1」という名前で保存されている図形を，配列X%に取り込む命令です。

③ BLOAD命令は，本来フロッピーディスクに保存されているマシン語のプログラムをメモリーにロードする命令ですが，ここでは図形データを扱うために利用しています。この命令では，プログラムをロードするときのメモリーの開始位置を指定する必要があります。そこで，図形を入れるための配列として定義したX%のメモリー上の開始位置をVARPTR関数を使って指定します。

④ 行番号130は，フロッピーディスクから配列X%に取り込んだ図形を，座標(300, 100)の位置に表示します。

設問 34-5　プログラムが正しくキーインできているかどうか確認した後，プログラムを実行して下さい。

＜実行結果例＞



〔実行後の説明〕

① 設問34-2で描いた図形が表示されます。

② このように，GET@とBSAVE，BLOADとPUT@といった命令でグラフィック図形のセーブとロードができます。これを使って，グラフィック画面全部をフロッピーディスクにセーブしておくこともできます。ただし，グラフィック画面全部をGETするための配列は大きすぎて定義することはできません。したがって，画面を分割してセーブしなければなりません。

【 命令のまとめ 】

① BSAVE命令の一般形式と書き方例を次に示します。

<一般形式>

BSAVE ファイルディスクリプタ，開始番地，長さ

・メモリー上のマシン語プログラムをフロッピーディスクにセーブします。

・ファイルディスクリプタで，「ドライブ番号」や「ファイル名」を指定します。

・開始番地は，セーブするマシン語プログラムが格納されているメモリー上の初めの番地を指定します。

・長さは，セーブするマシン語プログラムのバイト数を指定します。

<書き方例>

① BSAVE “2：ABCD.MAC”，&HC000，&HFF

これは，メモリー上のC000(16進)番地から格納されている255バイト(&HFF=255)のマシン語のプログラムを，ドライブ2のフロッピーディスクへ「ABCD.MAC」という名前で保存します。

<補　足>

・このテキストでは「BSAVE命令」が内容の種類に関係なく指定されたメモリーの内容をフロッピーディスクへセーブする機能を利用して，数値型配列内の値をフロッピーディスクへセーブします。したがって，セーブする配列のメモリー上の格納開始位置や長さを調べてから行います。

② BLOAD命令の一般形式と書き方例を次に示します。

＜一般形式＞

BLOAD ファイルディスクリプタ，開始番地

・フロッピーディスクにセーブしたマシン語プログラムを，メモリーにロードします。

・ファイルディスクリプタは，セーブするときに指定したものと同じものを指定します。

・開始番地は，マシン語プログラムをロードするメモリー上の初めの番地を指定します。

＜書き方例＞

① BLOAD "2：PROG.MAC"，&HE000

これは，ドライブ2のフロッピーディスクに「PROG.MAC」という名前でセーブしてあるマシン語のプログラムを，メモリー上のE000(16進)番地からロードします。

＜補　足＞

・このテキストでは，フロッピーディスクにセーブした図形の値を，配列変数にロードします。配列変数のメモリー上の開始位置は，VARPTR関数で簡単にわかります。ただし，配列変数はロードする前に必ず定義しておかなければなりません。

③ VARPTR関数の一般形式と書き方例を次に示します。

＜一般形式＞

VARPTR(変数名)

・変数名で指定された変数の中の値が格納されているメモリー上の番地を調べます。

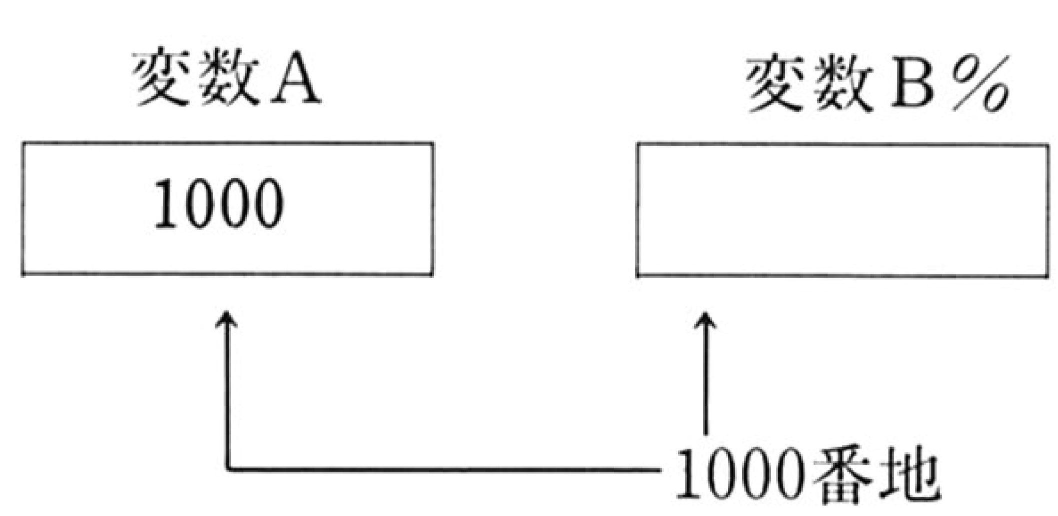
・配列変数を調べる場合は，添字も指定します。

・数値型変数の場合は，値そのものの番地を調べますが，文字列変数の場合は，ストリングディスクリプタと呼ばれる，文字列の長さや番地等の情報を納めたデータの番地を調べます。

＜書き方例＞

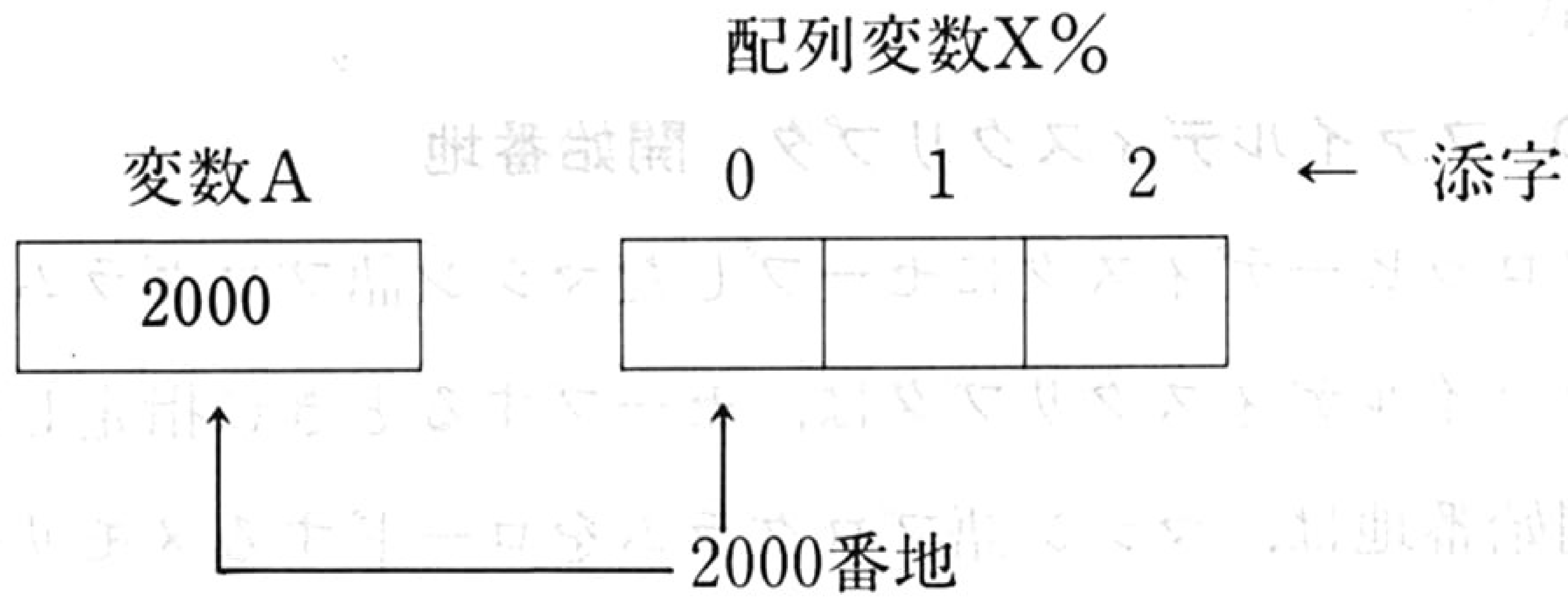
① A＝VARPTR(B%)

これは，変数B%の中の値がメモリー上の何番地から格納されているかを調べます。



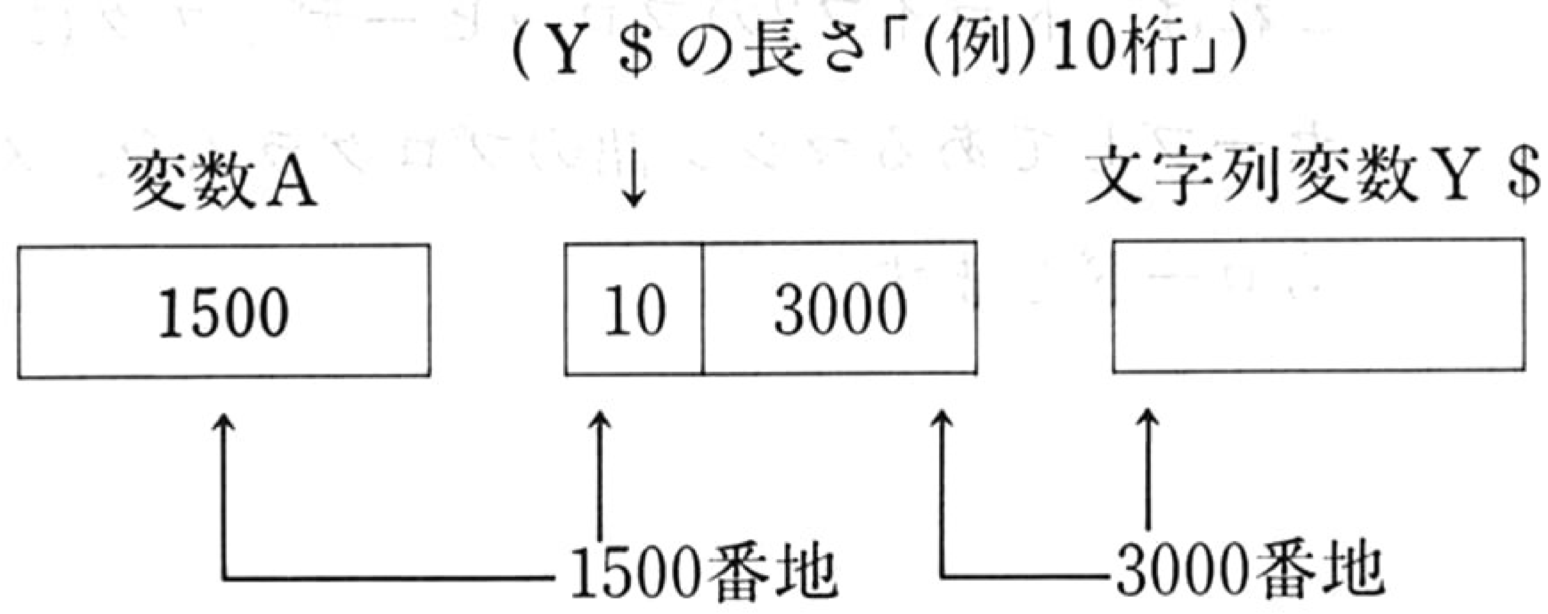
② A＝VARPTR(X％(0))

これは，配列変数X％の一番初めの要素が格納されている番地を調べます。



③ A＝VARPTR(Y＄)

これは，文字列変数Y＄の中の文字列の長さと格納番地を調べます。



第3章　ディスクコマンドとユーティリティ

ここでは，フロッピーディスクを使っていく上で必要なディスクコマンドとディスクユーティリティの使い方を学習します。

3.1　ディスクコマンドの使い方

今までの学習ではフロッピーディスクの使い方のうち，主にデータを記録したり読んだりあるいはメモリーにキーインしたプログラムを保存したり読み込んだりしました。ここでは，記録してあるプログラムの名前を調べたり変更したり，あるいはデータを変更できないように保護したりするときに必要になるフロッピーディスクの操作を行うためのコマンドの使い方について説明します。

3.1.1　ディレクトリリストの表示と印字（FILES／LFIES）

フロッピーディスクにどんなプログラムやデータファイルが記録されているかを調べるために「FILES」,「LFILES」というコマンドがあります。これらはフロッピーディスクの「ファイル管理領域」に記録されている情報(プログラムの名前や大きさ等)を画面に表示したり，プリンタに印字したりするコマンドです。

問題　35　ドライブ2のフロッピーディスクに何が記録されているか調べて下さい。

設問 35-1　FILESコマンドを実行して，ドライブ 2 のフロッピーディスクのディレクトリを画面に表示して下さい。

<実行結果例>

```
FILES 2
SEISEK MST 7     MONDAI.1    1     MONDAI.2    1     MONDAI.2-X 1     MONDAI.3    2
MONDAI.4    1    MONDAI.4-X 2      MONDAI.K1   1     MONDAI.K4   2    MONDAI.K3   2
JYUSHO DAT 13    JYUSHO IDX 1      MONDAI.5    1     MONDAI.6    2    MONDAI.7    1
MONDAI.8    2    MONDAI.9    1     MONDAI.12   1     MONDAI.13   1    MONDAI.16   1
MONDAI.20   1    MONDAI.21   1     MONDAI.27   1     MONDAI.28   1    MONDAI.29   1
MONDAI.33   1    GAMEN *1    1     MONDAI.34   1
Ok
```

〔実行後の説明〕

① ドライブ2のフロッピーディスクに記録されているプログラムやデータファイルが，一覧表形式(ディレクトリ)で表示されます。

② ディレクトリリストは，ファイル名やプログラム名とそれを修飾する拡張子の表示と，ファイルやプログラムの大きさの表示を行います。

③ ファイル名やプログラム名と拡張子の間の文字は区切り記号と呼ばれ，この区切り記号がピリオド(.)の場合は，そのファイルがバイナリー形式で保存されたプログラムであることを表しています。また，区切り記号が空白の場合は，そのファイルがアスキー形式で保存されたプログラムであるか，もしくはデータファイルであることを表しています。なお，区切り記号がアスタリスク(*)の場合は，そのファイルがマシン語のプログラムであることを表しています。

④ 拡張子の右側に表示されている数値は，ファイルの大きさをクラスタ(「1.4.1 フロッピーディスクの構造」を参照)単位で表しています。

設問 35-2 LFILESコマンドを実行して，ドライブ 2 のフロッピーディスクのディレクトリをプリンタに印字して下さい。

＜実行結果例＞

```
LFILES 2
Ok
```

(画面の結果例)

(プリンタの印字例)

```
SEISEK MST 7     MONDAI.1    1    MONDAI.2    1    MONDAI.2-X 1    MONDAI.3    2
MONDAI.4    1    MONDAI.4-X 2     MONDAI.K1   1    MONDAI.K4   2    MONDAI.K3   2
JYUSHO DAT 13    JYUSHO IDX 1     MONDAI.5    1    MONDAI.6    2    MONDAI.7    1
MONDAI.8    2    MONDAI.9    1    MONDAI.12   1    MONDAI.13   1    MONDAI.16   1
MONDAI.20   1    MONDAI.21   1    MONDAI.27   1    MONDAI.28   1    MONDAI.29   1
MONDAI.33   1    GAMEN *1    1    MONDAI.34   1
```

〔実行後の説明〕

① 設問35-1で画面に表示されたディレクトリと同じものがプリンタに印字されます。

【 命令のまとめ 】

FILESおよびLFILESコマンドの一般形式と書き方例を次に示します。

<一般形式>

FILES ドライブ番号

- ドライブ番号で指定したドライブに入っているフロッピーディスクのディレクトリリストを画面に表示します。
- ドライブ番号を省略すると「1」を指定したときと同じ結果になります。

LFILES ドライブ番号

- ドライブ番号で指定したドライブに入っているフロッピーディスクのディレクトリリストをプリンタに印字します。
- ドライブ番号を省略すると「1」を指定したときと同じ結果になります。

<書き方例>

① FILES

これは，ドライブ1のフロッピーディスクのディレクトリリストを画面に表示します。

② FILES 2

これは，ドライブ2のフロッピーディスクのディレクトリリストを画面に表示します。

③ LFILES

これは，ドライブ1のフロッピーディスクのディレクトリリストをプリンタに印字します。

④ LFILES 2

これは，ドライブ2のフロッピーディスクのディレクトリリストをプリンタに印字します。

3.1.2　プログラム名またはデータファイル名の変更(NAME)

通常，プログラムやデータファイルに一度付けた名前を変更することはあまりしませんが，特別の目的で使用するために名前を変更することがあります。

問題　36　ドライブ2のフロッピーディスクに入っている，「GAMEN.1」というマシン語プログラムを「ZUKEI.1」という名前に変更して下さい。

【 要点の説明 】

フロッピーディスクに入っているプログラムやデータファイルの名前を変更するには，「NAME」という命令を使います。

NAME命令は，変更するファイルの名前と変更後の名前を指定します。

設問 36-1　NAME命令を実行して，「GAMEN.1」を「ZUKEI.1」という名前に変更して下さい。

<実行結果例>

```
NAME "2:GAMEN.1" AS "2:ZUKEI.1"
Ok
```

〔実行後の説明〕

① ファイルディスクリプタの中で指定する「.」は，ファイル名と拡張ファイル名を区切るための記号であり，ディレクトリリスト中に表示される「.」とは意味が異なります。

設問 36-2　ドライブ 2 のフロッピーディスクのディレクトリリストを画面に表示して，名前の変更がうまくいったかどうかを確認して下さい。

<実行結果例>

```
FILES 2
SEISEK MST 7    MONDAI.1   1    MONDAI.2   1    MONDAI.2-X 1    MONDAI.3   2
MONDAI.4   1    MONDAI.4-X 2    MONDAI.K1  1    MONDAI.K4  2    MONDAI.K3  2
JYUSHO DAT 13   JYUSHO IDX 1    MONDAI.5   1    MONDAI.6   2    MONDAI.7   1
MONDAI.8   2    MONDAI.9   1    MONDAI.12  1    MONDAI.13  1    MONDAI.16  1
MONDAI.20  1    MONDAI.21  1    MONDAI.27  1    MONDAI.28  1    MONDAI.29  1
MONDAI.33  1    ZUKEI *1   1    MONDAI.34  1
Ok
```

【 命令のまとめ 】

NAME命令の一般形式と書き方例を次に示します。

<一般形式>

NAME 旧ファイルディスクリプタ AS 新ファイルディスクリプタ

・旧ファイルディスクリプタで指定したファイルまたはプログラムの名前を，新ファイルディスクリプタで指定したファイルまたはプログラムの名前に変更します。

・これから変更しようとするファイルまたはプログラムの名前を旧ファイルディスクリプタとして次の形式で指定します。

"ドライブ番号：旧ファイル名.拡張ファイル名"

・新しい名前を新ファイルディスクリプタとして次の形式で指定します。

"ドライブ番号：新ファイル名.拡張ファイル名"

<書き方例>

① NAME "PROG-1" AS "PROG-A"

これは，ドライブ1のフロッピーディスクに既に記録されている「PROG-1」というプログラムの名前を「PROG-A」という名前に変更します。

② NAME "2：KYUYO" AS "2：KYU-A1"

これは，ドライブ2のフロッピーディスクに既に記録されている「KYUYO」というデータファイルの名前を「KYU-A1」という名前に変更します。

3.1.3 プログラムまたはデータファイルの保護（SET）

フロッピーディスクに記録したプログラムやデータファイルを誤って消してしまったり，変更してはいけないファイルのデータを誤って変えてしまったりすることがあります。このようなことを未然に防ぐためにファイルやプログラムを保護しておくことができます。

問題 37 ドライブ2に入っているフロッピーディスクを，書き込みを行ったら必ず正しく記録できたかどうかを確かめるようにした後，「MONDAI.27」というプログラムを保護するようにしてください。

【説　明】

フロッピーディスクには，「リードアフターライト(書き込み後読み込み)」と「プロテクト(保護)」という二つの属性を付けることができます。

リードアフターライト(「R」属性)は，データをフロッピーディスクに書き込んだ後にそのデータを読み込んで，正しくデータの書き込みが行われたかどうかが確認されるようになります。

プロテクト(「P」属性)は，誤ってデータを消したり，プログラムを同じ名前でセーブしたために大事なプログラムが消えてしまったりということがないように，ファイルやフロッピーディスク自体を書き込みから保護する属性です。

設問 37-1　ドライブ 2 のフロッピーディスクに，リードアフターライト(R)の属性をセットして下さい。

<実行結果例>

```
set 2,"R"
Ok
```

〔実行後の説明〕

① ドライブ2のフロッピーディスクに「R」属性がセットされます。

② 属性のリセット(解除)は，“R”か“P”以外の文字を使ってSET命令を実行します。

設問 37-2　「MONDAI.27」というプログラムに，プロテクト(P)の属性をセットして下さい。

<実行結果例>

```
set "2:MONDAI.27","P"
Ok
```

〔実行後の説明〕

① ドライブ2のフロッピーディスクに入っている「MONDAI.27」というプログラムに，「P」属性がセットされます。

【 命令のまとめ 】

SET命令の一般形式と書き方例を次に示します。

<一般形式>

SET ｛ドライブ番号 ／ ファイルディスクリプタ ／ #ファイル番号｝，"属性"

・属性には「P」と「R」の2種類があり，それぞれ次の働きを行います。

「P」は書き込みができないようにします。

「R」はデータを記録した後に，必ず正しく記録されたかどうか確認します。

(注) 「P」,「R」は必ず大文字でなければなりません。また,この属性を解除する場合には,「P」と「R」以外の文字を指定します。

・属性の付け方には，ドライブ番号を指定してフロッピーディスク自身にセットする方法と，ファイルディスクリプタやファイル番号（ファイルがオープンされているとき)を指定してファイル単位にセットする方法があります。

<書き方例>

① SET 2，"P"

これは,ドライブ2のフロッピーディスクに記録してあるプログラムやデータファイルをすべて変更できない状態にします。

② SET "2：SHOHIN"，"R"

これは，ドライブ2のフロッピーディスクに記録してある「SHOHIN」というデータファイルへデータを記録したときに，必ず正しく記録できたかどうかを確認するようにします。

③ SET "2：PROG-1"，"P"

これは，ドライブ2のフロッピーディスクに記録してある「PROG-1」というプログラムを変更できない状態にします。

④ SET#1，"P"

これは，ファイル番号1のファイルを変更できない状態にします。

ただし，これはプログラムの中で用います。

3.1.4 プログラムまたはデータファイルの消去(KILL)

既にフロッピーディスクに記録してあるプログラムやデータファイルのうち，不要になったものを消去することができます。

問題　38 ドライブ2のフロッピーディスクに入っている「MONDAI.27」という名前のプログラムを消去して下さい。

【 要点の説明 】

「KILL」という命令を使って，不要になったファイルを消去することができます。この命令は，ファイル単位で消去を行います。したがって，一枚のフロッピーディスクに入っているすべてのファイルを消去する場合は，全部のファイル名を繰り返し指定して実行しなければなりません。(すべてのファイルを消去する場合は，そのフロッピーディスクに対してフォーマットを行った方が便利です)

設問 38-1 ドライブ 2 のフロッピーディスクに入っている「MONDAI.27」という名前のプログラムを消去して下さい。

<実行結果例>

```
kill "2:MONDAI.27"
File write protected
Ok
```

〔実行後の説明〕

① 「MONDAI.27」というプログラムは，設問37-2でプロテクトの属性をつけたため変更や消去ができないようになっています。このために「File write protected」のエラーが発生しています。

② プロテクトされているファイルを意識的に消去する場合は，次のように「SET命令」を使って一度プログラムやファイルを変更できる状態にしてから行います。

SET “2：MONDAI.27”，““

設問 38-2 「MONDAI.27」の属性をリセットして，変更や消去ができる状態にして下さい。

＜実行結果例＞

```
set "2:MONDAI.27",""
Ok
```

設問 38-3 再び設問38-1を実行して下さい。

＜実行結果例＞

```
kill "2:MONDAI.27"
Ok
```

設問 38-4 ドライブ 2 のフロッピーディスクのディレクトリを表示して，「MONDAI.27」というプログラムが消去されていることを確認して下さい。

＜実行結果例＞

```
FILES 2
SEISEK MST 7    MONDAI.1   1    MONDAI.2   1    MONDAI.2-X 1    MONDAI.3   2
MONDAI.4   1    MONDAI.4-X 2    MONDAI.K1  1    MONDAI.K4  2    MONDAI.K3  2
JYUSHO DAT 13   JYUSHO IDX 1    MONDAI.5   1    MONDAI.6   2    MONDAI.7   1
MONDAI.8   2    MONDAI.9   1    MONDAI.12  1    MONDAI.13  1    MONDAI.16  1
MONDAI.20  1    MONDAI.21  1    MONDAI.28  1    MONDAI.29  1    MONDAI.33  1
ZUKEI *1   1    MONDAI.34  1
Ok
```

【 命令のまとめ 】

KILL命令の一般形式と書き方例を次に示します。

<一般形式>

KILL ファイルディスクリプタ

・ファイルディスクリプタで指定したプログラムまたはファイルを消去します。

<書き方例>

① KILL "FUYOU"

これは，ドライブ1のフロッピーディスクに書かれている「FUYOU」という名前のファイルを消去します。

② KILL "2：PROG"

これは，ドライブ2のフロッピーディスクに書かれている「PROG」という名前のプログラムを消去します。

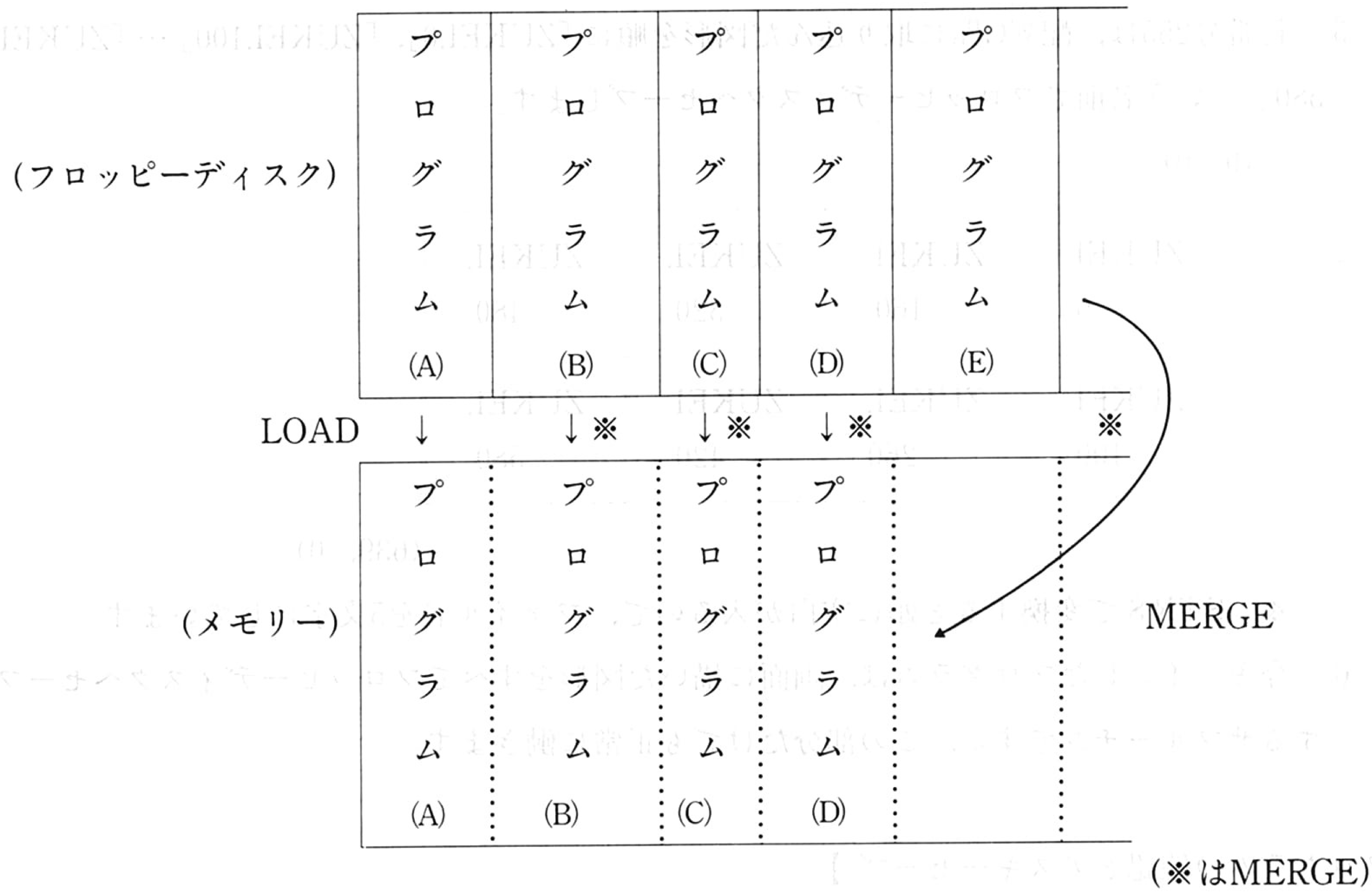
3.1.5 プログラムとプログラムの継ぎ合わせ(MERGE)

N_{88}DISK-BASICでは，既にフロッピーディスクにアスキー形式で保存してあるプログラムをメモリーのプログラムに継ぎ合わせることができます。

問題 39 タートルを使って描いた図形をフロッピーディスクへセーブする(ビューポート内)問題33のプログラムを，画面全部の図形がセーブできるように改造して下さい。

【 要点の説明 】

大きなプログラムを作る場合，初めから終わりまでを一度に作ることは大変な作業になり，間違いも多くなります。そこで，作ろうとするプログラムを，処理機能ごとにサブルーチンの形式にしてしまいます。そして，おのおののサブルーチンを一つ一つのプログラムにしてしまえば，他の人と分担してプログラムを作ることができます。このようにして作った一つ一つのプログラムを，最後に継ぎ合わせ(MERGE)機能を利用して一つのプログラムにします。この方法を利用することにより，プログラムの作成を容易にしたり，作成期間の短縮を図ったり，プログラムの間違いを少なくしたりすることができます。



ここでは，問題33のプログラムの改造部分を一つのサブルーチン(プログラム)として，フロッピーディスクにアスキーセーブしておき，問題34のプログラムと継ぎ合わせます。

設問 39-1 メモリーをクリアしてから，次のプログラムをキーインして下さい。

<キーインするプログラム>

```
102 DIM G%(3002)
252  FOR X=0 TO 480 STEP 160
253    FOR Y=0 TO 100 STEP 100
254      GET@(X,Y)-STEP(159,99),G%
255      BSAVE "2:ZUKEI"+STR$(X+Y),VARPTR(G%(0)),6006
256  NEXT Y,X
257  END
```

〔キーイン後の説明〕

① キーインしたプログラムは，グラフィック画面を8分割してフロッピーディスクにセーブするものです。

② 行番号102は，8分割したグラフィック画面の図形を取り込んでおくための配列G%を定義しています。

③ 行番号252と253は，X座標とY座標が8等分になるように分割しながら繰り返す処理です。

④ 行番号254は，分割された画面の各図形を配列G%に取り込みます。

⑤ 行番号255は，配列G%に取り込んだ図形を順に「ZUKEI.0」,「ZUKEI.100」…「ZUKEI.580」という名前でフロッピーディスクへセーブします。

(0, 0)

ZUKEI. 0	ZUKEI. 160	ZUKEI. 320	ZUKEI. 480
ZUKEI. 100	ZUKEI. 260	ZUKEI. 420	ZUKEI. 580

(639, 0)

※ STR＄で変換すると頭に空白が入るので，ファイル名を5文字にしています。

⑥ 今キーインしたプログラムは，画面に描いた図形をすべてフロッピーディスクへセーブするサブルーチンですが，この部分だけでも正常に働きます。

【 プログラムの確認とアスキーセーブ 】

設問 39-2 プログラムのリストを画面に表示して,正しくキーインできているかどうか確認して下さい。

<実行結果例>

```
102 DIM G%(3002)
252  FOR X=0 TO 480 STEP 160
253    FOR Y=0 TO 100 STEP 100
254      GET@(X,Y)-STEP(159,99),G%
255      BSAVE "2:ZUKEI"+STR$(X+Y),VARPTR(G%(0)),6006
256  NEXT Y,X
257  END
```

設問 39-3 正しくキーインできていたら,このプログラムをドライブ 2 のフロッピーディスクへ「SUBPRG.39」という名前でアスキーセーブして下さい。

<実行結果例>

```
save "2:SUBPRG.39",a
Ok
```

〔実行後の説明〕

① ドライブ2のフロッピーディスクに「SUBPRG.39」という名前のプログラムがアスキーセーブされました。

② アスキーセーブは，シーケンシャルファイルと同じ形式でフロッピーディスク上に保存されます。

設問 39-4 ドライブ 2 のフロッピーディスクのディレクトリを表示して下さい。

<実行結果例>

```
FILES 2
SEISEK MST 7     MONDAI.1    1    MONDAI.2    1    MONDAI.2-X 1    MONDAI.3    2
MONDAI.4   1     MONDAI.4-X  2    MONDAI.K1   1    MONDAI.K4  2    MONDAI.K3   2
JYUSHO DAT 13    JYUSHO IDX  1    MONDAI.5    1    MONDAI.6   2    MONDAI.7    1
MONDAI.8   2     MONDAI.9    1    MONDAI.12   1    MONDAI.13  1    MONDAI.16   1
MONDAI.20  1     MONDAI.21   1    SUBPRG 39   1    MONDAI.28  1    MONDAI.29   1
MONDAI.33  1     ZUKEI *1    1    MONDAI.34   1
Ok
```

〔実行後の説明〕

① アスキーセーブされたプログラムは，ディレクトリリストを表示したときに他のプログラムと違ってプログラム名と拡張ファイル名の間に「.」が表示されません。

【 ロードとマージ 】

設問 39-5 ドライブ 2 のフロッピーディスクから，「MONDAI.34」という名前のプログラムをメモリーにロードして下さい。

<実行結果例>

```
load "2:MONDAI.34"
Ok
```

設問 39-6 プログラムのリストをプリンタに印字して下さい。

<プログラムリスト>

```
100 SCREEN 0,0:CMD TURTLE "HT":CMD CLS 3
102 DIM G%(978)
```

（つづく）

```
104 VIEW (401,1)-(500,50),,7
110 POINT (50,25)
120 CMD TURTLE "ST":TSW=1:PSW=1
130 CLS:N$=INPUT$(1):N%=VAL(N$)
132 IF N$>"9" THEN GOSUB *GSC:GOTO 130
140 ON N% GOSUB *N1,*N2,*N3,*N4,*N5,*N6,*N7,*N8,*N9
150 GOTO 130
160 *N1:CMD TURTLE "HD225 FD1":RETURN
170 *N2:CMD TURTLE "HD180 FD1":RETURN
180 *N3:CMD TURTLE "HD135 FD1":RETURN
190 *N4:CMD TURTLE "HD270 FD1":RETURN
200 *N5:CMD TURTLE "FD1":RETURN
210 *N6:CMD TURTLE "HD90 FD1":RETURN
220 *N7:CMD TURTLE "HD315 FD1":RETURN
230 *N8:CMD TURTLE "HD0 FD1":RETURN
240 *N9:CMD TURTLE "HD45 FD1":RETURN
250 *GSC:' ------ ｸﾞﾗﾌｨｯｸ ｻﾌﾞｺﾏﾝﾄﾞ ------
251  IF N$><"E" THEN 260
252     VIEW (0,0)-(639,199)
253     GET@ (401,1)-(500,50),G%
254     BSAVE "2:GAMEN.1",VARPTR(G%(0)),2*(978+1)
255     END
260  IF N$><"T" THEN 290
270     IF TSW=1 THEN T$="HT" ELSE T$="ST"
280     CMD TURTLE T$:TSW=1-TSW:RETURN
290  IF N$><"F" THEN 320
300     INPUT "ｽｽﾒﾙｶｽﾞﾊ";N
310     CMD TURTLE "FD(N)":RETURN
320  IF N$><"B" THEN 350
330     INPUT "ｺｳﾀｲｻｾﾙｶｽﾞﾊ";N
340     CMD TURTLE "BK(N)":RETURN
350  IF N$><"L" THEN 370
360     CMD TURTLE "LT1":RETURN
370  IF N$><"R" THEN 390
380     CMD TURTLE "RT1":RETURN
390  IF N$><"C" THEN 420
400     INPUT "ｶﾗｰﾊﾞﾝｺﾞｳ";N
410     CMD TURTLE "PC(N)":RETURN
420  IF N$><"P" THEN 450
430     IF PSW=1 THEN P$="PU" ELSE P$="PD"
440     CMD TURTLE P$:PSW=1-PSW:RETURN
450  IF N$><"M" THEN 480
460     INPUT "ｲﾄﾞｳｽﾙｲﾁﾊ    x,y  ";X,Y
470     CMD TURTLE "MV(X),(Y)":RETURN
480  IF N$><"@" THEN RETURN
490     INPUT "ｷｮｳｶｲｾﾝ ﾉ ｲﾛ";C2
500     IF C2<0 OR C2>7 THEN 490
510     INPUT "ﾇﾘﾂﾌﾞｽ ｲﾛ";C1
520     IF C1<0 OR C1>7 THEN 540
530        PAINT STEP(0,0),C1,C2:RETURN
540     INPUT "ﾀｲﾙﾊﾟﾀｰﾝ ﾉ ﾗｲﾝｽｳ";L
550     TILE$=""
560     FOR I=1 TO L
570       INPUT "ﾌﾟﾚｰﾝ1 ﾉ ﾊﾟﾀｰﾝ (xx) ";A$:A$=CHR$(VAL("&H"+A$))
580       INPUT "ﾌﾟﾚｰﾝ2 ﾉ ﾊﾟﾀｰﾝ (xx) ";B$:B$=CHR$(VAL("&H"+B$))
590       INPUT "ﾌﾟﾚｰﾝ3 ﾉ ﾊﾟﾀｰﾝ (xx) ";C$:C$=CHR$(VAL("&H"+C$))
600       TILE$=TILE$+A$+B$+C$
610     NEXT I
620     PAINT STEP(0,0),TILE$,C2:RETURN
```

設問 39-7 いまメモリーに記憶しているプログラムとドライブ 2 のフロッピーディスクに保存してある「SUBPRG.39」というプログラムをMERGEコマンドを使って継ぎ合わせて下さい。

<実行結果例>

```
MERGE "2:SUBPRG.39"
Ok
```

設問 39-8 プログラムのリストをプリンタに印字して,継ぎ合わせがうまくできているかどうか確認して下さい。

<確認用リスト>

```
100 SCREEN 0,0:CMD TURTLE "HT":CMD CLS 3
102 DIM G%(3002)
104 VIEW (401,1)-(500,50),,7
110 POINT (50,25)
120 CMD TURTLE "ST":TSW=1:PSW=1
130 CLS:N$=INPUT$(1):N%=VAL(N$)
132 IF N$>"9" THEN GOSUB *GSC:GOTO 130
140 ON N% GOSUB *N1,*N2,*N3,*N4,*N5,*N6,*N7,*N8,*N9
150 GOTO 130
160 *N1:CMD TURTLE "HD225 FD1":RETURN
170 *N2:CMD TURTLE "HD180 FD1":RETURN
180 *N3:CMD TURTLE "HD135 FD1":RETURN
190 *N4:CMD TURTLE "HD270 FD1":RETURN
200 *N5:CMD TURTLE "FD1":RETURN
210 *N6:CMD TURTLE "HD90 FD1":RETURN
220 *N7:CMD TURTLE "HD315 FD1":RETURN
230 *N8:CMD TURTLE "HD0 FD1":RETURN
240 *N9:CMD TURTLE "HD45 FD1":RETURN
250 *GSC:' ------ ｸﾞﾗﾌｨｯｸ ｻﾌﾞｺﾏﾝﾄﾞ ------
251  IF N$><"E" THEN 260
252  FOR X=0 TO 480 STEP 160
253    FOR Y=0 TO 100 STEP 100
254      GET@(X,Y)-STEP(159,99),G%
255      BSAVE "2:ZUKEI"+STR$(X+Y),VARPTR(G%(0)),6006
256  NEXT Y,X
257  END
260  IF N$><"T" THEN 290
270     IF TSW=1 THEN T$="HT" ELSE T$="ST"
280     CMD TURTLE T$:TSW=1-TSW:RETURN
290  IF N$><"F" THEN 320
300     INPUT "ｽｽﾒﾙｶｽﾞﾊ";N
310     CMD TURTLE "FD(N)":RETURN
320  IF N$><"B" THEN 350
330     INPUT "ｺｳﾀｲｻｾﾙｶｽﾞﾊ";N
340     CMD TURTLE "BK(N)":RETURN
350  IF N$><"L" THEN 370
360     CMD TURTLE "LT1":RETURN
370  IF N$><"R" THEN 390
```

(つづく)

```
380      CMD TURTLE "RT1":RETURN
390   IF N$><"C" THEN 420
400      INPUT "ｶﾗｰﾊﾞﾝｺﾞｳ";N
410      CMD TURTLE "PC(N)":RETURN
420   IF N$><"P" THEN 450
430      IF PSW=1 THEN P$="PU" ELSE P$="PD"
440      CMD TURTLE P$:PSW=1-PSW:RETURN
450   IF N$><"M" THEN 480
460      INPUT "ｲﾄﾞｳｽﾙｲﾁﾊ    x,y ";X,Y
470      CMD TURTLE "MV(X),(Y)":RETURN
480   IF N$><"@" THEN RETURN
490      INPUT "ｷｮｳｶｲｾﾝ ﾉ ｲﾛ";C2
500      IF C2<0 OR C2>7 THEN 490
510      INPUT "ﾇﾘﾂﾌﾞｽ ｲﾛ";C1
520      IF C1<0 OR C1>7 THEN 540
530         PAINT STEP(0,0),C1,C2:RETURN
540      INPUT "ﾀｲﾙﾊﾟﾀｰﾝ ﾉ ﾗｲﾝｽｳ";L
550      TILE$=""
560      FOR I=1 TO L
570        INPUT "ﾌﾟﾚｰﾝ1 ﾉ ﾊﾟﾀｰﾝ (xx) ";A$:A$=CHR$(VAL("&H"+A$))
580        INPUT "ﾌﾟﾚｰﾝ2 ﾉ ﾊﾟﾀｰﾝ (xx) ";B$:B$=CHR$(VAL("&H"+B$))
590        INPUT "ﾌﾟﾚｰﾝ3 ﾉ ﾊﾟﾀｰﾝ (xx) ";C$:C$=CHR$(VAL("&H"+C$))
600        TILE$=TILE$+A$+B$+C$
610      NEXT I
620      PAINT STEP(0,0),TILE$,C2:RETURN
```

〔実行後の説明〕

① メモリー上のプログラムとフロッピーディスクのプログラムとが継ぎ合わされています。

② 行番号が同じ場合，フロッピーディスク上のプログラムの命令に置き換わります。

③ このプログラムでは，行番号104でビューポートを設定しているために画面いっぱいに図形を描くことができません。

設問 39-9 行番号104を削除して下さい。

<実行結果例>

```
104
または
DELETE 104
```

〔実行後の説明〕

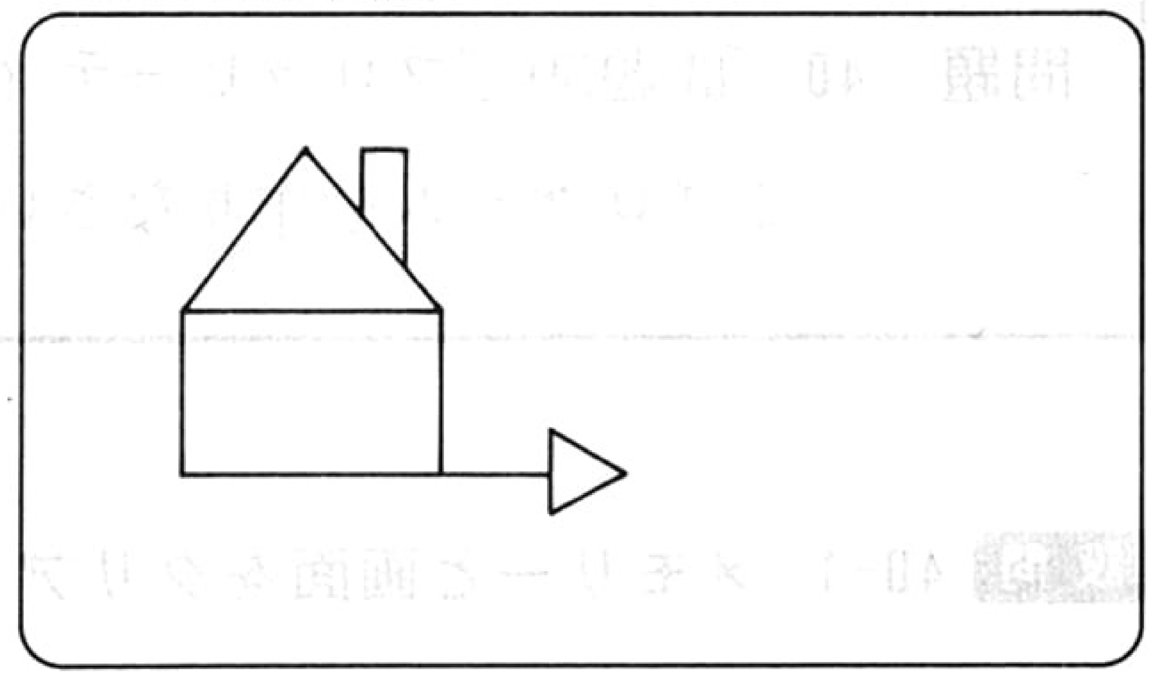
① これで，グラフィック画面にタートルで図形を描いた後，それをフロッピーディスクへセーブするプログラムができあがったことになります。

設問 39-10 プログラムを実行して，グラフィック画面に好きな図形を描いた後その画面全部をフロッピーディスクにセーブして下さい。

〔操作説明〕

・タートルで図形を描いた後，「E」を入れて
　フロッピーディスクへセーブします。

＜実行結果例＞



設問 **39-11 このプログラムをドライブ 2 のフロッピーディスクへ「MONDAI.39」という名前で保存して下さい。**

＜実行結果例＞

```
save "2:MONDAI.39"
Ok
```

【 命令のまとめ 】

MERGEコマンドの一般形式と書き方例を次に示します。

＜一般形式＞

MERGE　ファイルディスクリプタ

・ファイルディスクリプタで指定したプログラムを，フロッピーディスクからメモリーにロードしますが，このとき，既にメモリーに記憶していたプログラムの消去は行いません。したがって，メモリーとフロッピーディスクの二つのプログラムが継ぎ合わせられるということになります。

・ファイルディスクリプタで指定するプログラムは，あらかじめアスキー形式で保存されたものでなければなりません。

・メモリーに記憶しているプログラムの中に，フロッピーディスクのプログラムの中の行番号と同じ行番号があるときは，その行番号の命令はフロッピーディスクからメモリーに移した方のプログラムの命令に変更されます。

＜書き方例＞

① MERGE "2：EX-50S"

これは，ドライブ2のフロッピーディスクにアスキー形式で保存してある「EX-50S」というプログラムをメモリーのプログラムに継ぎ合わせる例です。

問題　40　問題39でフロッピーディスクにセーブしたグラフィック画面をロードして表示するプログラムを作りなさい。

設問 40-1 メモリーと画面をクリアしてから，次のプログラムをキーインして下さい。

<キーインするプログラム>

```
100 CLS
110 DIM G%(3002)
120 FOR X=0 TO 480 STEP 160
130   FOR Y=0 TO 100 STEP 100
140     BLOAD "2:ZUKEI"+STR$(X+Y),VARPTR(G%(0))
150     PUT@ (X,Y),G%
160 NEXT Y,X
```

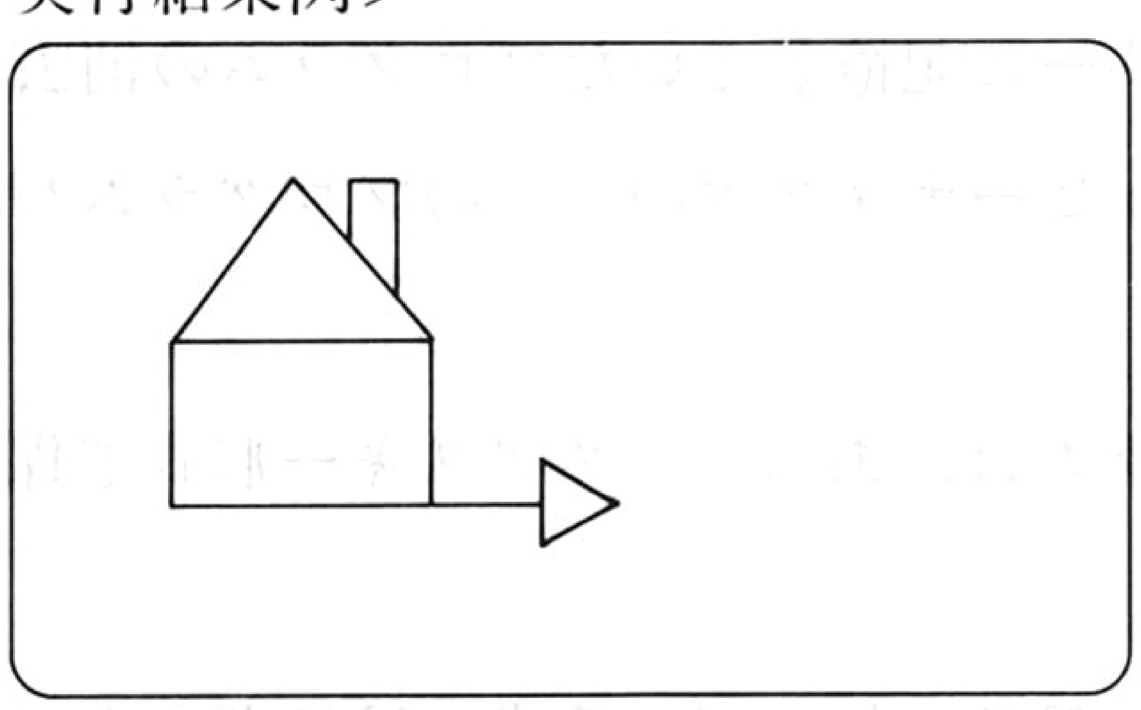
〔キーイン後の説明〕

① 行番号130で，フロッピーディスクから図形を配列G%へロードします。

② 行番号140は，配列にロードした図形を画面に表示します。

設問 40-2 プログラムを実行して，設問39-10でフロッピーディスクにセーブした図形をロードして画面に表示して下さい。

<実行結果例>



設問 40-3 このプログラムをドライブ 2 のフロッピーディスクへ「MONDAI.40」という名前で保存して下さい。

<実行結果例>

```
save "2:MONDAI.40"
Ok
```

3.2 ディスクユーティリティの使い方

フロッピーディスクを利用していくと，いろいろなときにフロッピーディスク自身をメンテナンス(保守)する必要がでてきます。上巻では，フロッピーディスクの最も基本的なメンテナンス作業として，フォーマッティングとバックアップの二つについて説明しました。この二つの作業は，N_{88}DISK-BASICのシステムディスクに用意されている「dskut2.n88」というユーティリティプログラムの中の機能を使って行いました。ここでは，「dskut2.n88」が持っているほかのユーティリティの機能とその操作の説明を行います。

3.2.1 ディスクユーティリティのメニュー

N_{88}DISK-BASICのシステムディスクには，「dskut2.n88」と「dskut1.n88」という二つのユーティリティプログラムが用意されています。「dskut1.n88」は，ミニフロッピーディスクが1ドライブの場合に使用するものです。ここでは，ミニフロッピーディスクのドライブが二つの場合に使用する「dskut2.n88」について説明します。

> **問題 41** N_{88}DISK-BASICのシステムディスクに入っている「dskut2.n88」というユーティリティプログラムを実行して，その機能と操作について学習して下さい。

設問 41-1 システムディスクをドライブ 1 にセットして「dskut2.n88」という名前のプログラムをロードして実行して下さい。

<実行結果例>

```
DISK UTILITY for PC-8801MK2   dskut2

        M  E  N  U
   COPY DISK
   CREATE SYSTEM DISK [N88]
   CREATE SYSTEM DISK [ N ]
   CREATE DATA DISK
   TRANSFER FILES
   SET UP INFO        [N88]
   SET UP INFO        [ N ]

Select a feature and press RETURN key.
```

〔実行後の説明〕

① このプログラムの実行は，先に「CMD CUT ⏎」を実行してから行います。(「Q&A Q—26」を参照)

② プログラムを実行するとメニュー画面が表示されます。

③ メニュー画面が表示されたら，カーソル移動キー(↑ ， ↓)を使って実行したい機能を選択して， ⏎ キーを押します。(選択した機能は，反転表示で示されます)

④ メニューで表示されている機能の概要は次のとおりです。

・COPY DISK
　フロッピーディスクのコピーを行い，バックアップシートを作ります。

・CREATE SYSTEM DISK [N88]
　N_{88}-DISK BASICのシステムディスクを作成します。

・CREATE SYSTEM DISK [N]
　DISK BASICのシステムディスクを作成します。

・CREATE DATA DISK
　データ記録専用のディスクを作成します。

・TRANSFER FILES
　ファイル単位のコピーを行ってバックアップシートを作ります。

・SET UP INFO [N88]
　カレンダ時計の年のセットを行ったり，N_{88}DISK-BASICを起動するとすぐに実行させたい処理がある場合に，これを定義したりします。

・SET UP INFO [N]
　DISK BASICを起動するとすぐに実行させたい処理がある場合に，これを定義します。

⑤ フロッピーディスクのコピー(バックアップ)とデータディスクの作成については，上巻を参照して下さい。なお，DISK-BASIC(N_{80}用)に関するものについては省略します。

3.2.2 N_{88}DISK-BASICシステムデイスクの作成

今までの学習では，システムディスクをドライブ1にセットして，学習で作成したプログラムやデータをドライブ2へ記録するようにしました。これは，N_{88}DISK-BASICを起動するためには，システムディスクが必要でかつプログラムやデータの記録を行うフロッピーディスクが必要になるからです。ところが，自分で作成したプログラムが大きくなったり扱うデータの量が増えたりしてくると，ドライブ1と2のどちらも必要になります。そこで，ドライブ1にはN_{88}DISK-BASICを起動するために必要な情報だけが記録されているシステムディスクを使うようにします。

ここでは，N_{88}DISK-BASICを起動するために必要な情報だけが記録されているシステムディスクを作成する方法を説明します。

設問 **41-2 カーソル移動キー(↓)を使って,「CREATE SYSTEM DISK[N88]」という機能を選択して ⏎ キーを押して下さい。**

<システムディスク作成のための操作手順>

メニュー画面から「CREATE SYSTEM DISK [N88]」という機能を選択して実行するときの操作手順を次表に示します。

番号	表示されるメッセージ	操　　作
①	[[[CREATE SYSTEM DISK]]] N88 DISK-BASIC Mount SYSTEM disk on drive. Drive number? ■	N_{88}DISK-BASICのシステムディスクが入っているドライブの番号を指定します。ここではドライブ1を指定します。
②	Mount NEW disk on drive. Drive number? ■	システムディスクにする新しいフロッピーディスクをセットし，そのドライブ番号を入力します。一般にはドライブ2を使います。
③	Do you need physical formatting(y/n)? ■ Format a disk on drive 2 Sure(y/n)? ■	システムディスクにしようとするフロッピーディスクを，まだ一度も使ったことがない場合は，「y」を入力します。一度でも使ったことがあれば「n」を入力します。 「n」を入力したときは，番号⑤のメッセージから実行が行われていきます。
④	Physical formatting. System formatting.	標準フロッピーディスクの場合はさらに次のメッセージが表示されます。 Formatting track x ∫ Formatting track xxx
⑤	Copying system.	
⑥	Completed. Ok	

この機能によって生成されたシステムディスクは，N_{88}DISK-BASIC を起動するために必要な情報だけを持っています。この情報の記録のために使われる領域は，4トラック分程度です。したがって、残りの領域はすべてプログラムやデータを記録するために使うことができますが，ユーティリティやデモプログラムおよび拡張命令などは実行できません。

3.2.3　5インチ，8インチフロッピーディスク間のコピー

「COPY　DISK」という機能を使うと，フロッピーディスクのバックアップをとることができます。しかし，この機能でバックアップをとることができるのは，5インチから5インチ，8インチから8インチというように同じ種類のフロッピーディスクの場合だけです。(同じ大きさのフロッピーディスクでも，片面倍密度と両面倍密度といった違いがある場合はコピーできません)

そこで，これらの異なるフロッピーディスク同士でもバックアップが取れるように，「TRANSFER　FILES」という機能が用意されています。

「COPY　DISK」では，同じ位置に同じ内容をそっくりそのままコピーしますが，「TRANSFER FILES」では，ファイルごとにコピーを行います。ですから，異なる種類のフロッピーディスク間でもコピーを行うことができるわけですが，記録容量の大きいフロッピーディスク(たとえば，8インチフロッピーディスク)から容量の小さいフロッピーディスク(たとえば，5インチフロッピーディスク)へコピーするときは容量の不足でコピーが完全にできない場合があります。この場合は，「Disk full」というエラーメッセージが表示されます。

設問 41-3　ディスクユーティリティを再び実行して，メニューの中から「TRANSFER-FILES」の機能を選択して ⏎ キーを押して下さい。

〔ファイル単位のコピーを行うための操作手順〕

番号	表示されるメッセージ	操　　作
①	[[[TRANSFER FILES]]] Transfer from ? ■	バックアップをとるフロッピーディスクをセットしてそのドライブ番号を入力します。一般にはドライブ1を使います。 片面倍密度フロッピーディスクの場合は，ドライブ番号の後に「S」をつけます。
②	to ? ■	バックアップ用のフロッピーディスクをセットしてそのドライブ番号を入力します。ここではドライブ2を使います。
③	Transfer from ■ to ■ Sure(y/n)? ■	フロッピーディスクをセットしたドライブと入力したドライブ番号を確認して，間違いがなければ「y」を，間違いがあれば「n」を入力します。 「n」を入力すると，①からやり直しになります。

番号	表示されるメッセージ	操　　作
④	Copying file: xxxxxxxxx　　1 -> 2 ⌇ Copying file: xxxxxxxxx　　1 -> 2	
⑤	Completed. Ok	

3.2.4　年の自動設定とオートスタート

PC-8801mkIIではシステムを起動した直後のカレンダ時計は「84年」になっています。

そこで，これを変更する場合はN_{88}DISK-BASICのシステムディスクに登録されているディスクユーティリティを使います。また，このディスクユーティリティはシステムを起動したときに特別のプログラムを自動的に実行するように定義することもできます。

設問 41-4　ディスクユーティリティを再び実行して，メニューの中から「SET UP INFO［N 88］」の機能を選択して ⏎ キーを押して下さい。

〔年の自動設定とオートスタートを行うための操作手順〕

番号	表示されるメッセージ	操　　作
①	[[[SET UP INFO]]] 　N88 DISK-BASIC Mount SYSTEM disk on drive. Drive number? ■	カレンダ時計の年号を変えたり，オートスタートを行ったりするシステムディスクをセットして，そのドライブ番号を入力します。
②	How many files (-1 or 0-15) ? ■	オートスタートを行うときに用意するバッファの数を入力します。起動時にその都度用意するバッファの数を決める場合は，「−1」を入力します。
③	Year for init (-1 or 0-99) ■	起動時のカレンダ時計の年を入力します。「84」でよい場合は「−1」を入力します。(84を入力してもよい)
④	Text for init ? ■	プログラムのオートスタートや起動直後に何か処理を行いたい場合にその命令を指定します。たとえば，起動直後にドライブ2の「PRG1」というプログラムをオートスタートさせたい場合は，run "2：PRG1" ⏎ とキーインします。

番号	表示されるメッセージ	操　作
⑤	Number of files to allocate is xx または Number of files is asked when starting up Year for init is xx Text for init is xxxxx	指定した内容が，確認のために表示されます。
⑥	Sure(y/n)? ■	⑤で表示された内容を確認して，正しければ「y」間違っていれば「n」を入力します。
⑦	Completed. Ok	

（注） ④で指定した命令が長すぎた場合は，次のようなメッセージを表示してブザーが鳴ります。この場合は，BASICの命令を検討して短くしてから再度入力してください。

```
Text can have a maximum of xxx characters
```

第4章　割り込みとエラー処理

ある処理を実行している最中に，一時的に他の異なる処理を行った後，再び元の処理を続けて行いたいという場合があります。このようにプログラムの実行中に，外部からの信号または指示によって一時的に異なる処理を行うことを「割り込み処理」といいます。

割り込み処理は，プログラムの実行中に特定のキーが押されたときや一定の時刻になったときあるいは通信回線から信号が入力されたときに発生します。したがって，プログラムではキーボードからの指示や時刻などを監視していなければ，割り込み処理を行うことはできません。ところが，主目的の処理を実行しているときに，プログラムがいつもキーボード等からの指示や時刻を見守っているのは，プログラムの効率が低下する原因になります。そこで，これを解決するためにPC-8801mkIIには機械自身（ハードウェア）がいつも割り込みの指示を見守っていて，その旨をプログラムに知らせてくれる「割り込み機能」があります。

ここでは，PC-8801mkIIが持っている割り込み機能について学習します。

4.1　割り込みの種類

N_{88}Disk-BASICが用意している割り込み機能には，次のものがあります。

- ・ファンクションキーによる割り込み　(ON KEY GOSUB)
- ・ストップキーによる割り込み　(ON STOP GOSUB)
- ・ヘルプキーによる割り込み　(ON HELP GOSUB)
- ・RS-232C通信回線による割り込み　(ON COM GOSUB)
- ・ライトペンによる割り込み　(ON PEN GOSUB)
- ・リアルタイム・タイマによる割り込み　(ON TIME$ GOSUB)

上に掲げたように，N_{88}Disk-BASICが割り込みとして処理を行うための要因には，キーボード，通信回線，ライトペン，リアルタイム・タイマの四つがあります。プログラムでは，これらのうちで必要な機能だけを定義します。

割り込み発生時に行う処理は，通常のサブルーチンと同様に作成しますが，プログラムの先頭（割り込みを行う前）で必ず割り込みが起こったときにどのサブルーチンを実行するのかということを定義する「ON 〈割り込み要因〉 GOSUB～」という命令を実行します。

プログラムでは，さらに割り込みの許可，保留，禁止を設定する「〈割り込み要因〉ON／OFF／STOP」という命令を実行して，割り込みの制御を行います。（たとえば，ファイルへデータを書き込んでいるときに，割り込みが入ってその処理を行うとデータが変わってしまうときなどは，割り込みを禁止しておかなければなりません）

4.1.1 ファンクションキーによる割り込み （ON KEY GOSUB）

これは，プログラムを実行している最中に，ファンクションキーが押されたとき割り込み機能が働くように定義するものです。

ファンクションキーは，[f･1]～[f･10]まで10個あります。したがって，ファンクションキーによる割り込みでは，[f･1]キーが押されたときAの処理，[f･2]キーが押されたときのBの処理といったように，各ファンクションキーに対して別々の割り込み処理を行うことができます。

問題 42 三角関数を使って円を描くプログラムを作った後，ファンクションキーによって色を変えられるようにして下さい。

・描く円の条件

中心　(300, 100)

半径　100

円周上の点の位置の求め方

X座標　X＝半径＊COS（角度＊3.14159/180)＋中心のX座標

Y座標　Y＝半径＊SIN（角度＊3.14159/180)/2＋中心のY座標

・ファンクションキーによる色の変更

[f･1]→　カラーコード　1　青色

[f･2]→　カラーコード　2　赤色

～

[f･7]→　カラーコード　7　白色

[f･8]～[f･10]は変更しない

【 プログラムの作成 】

設問 42-1 次のプログラムをキーインして下さい。

＜キーインするプログラム＞

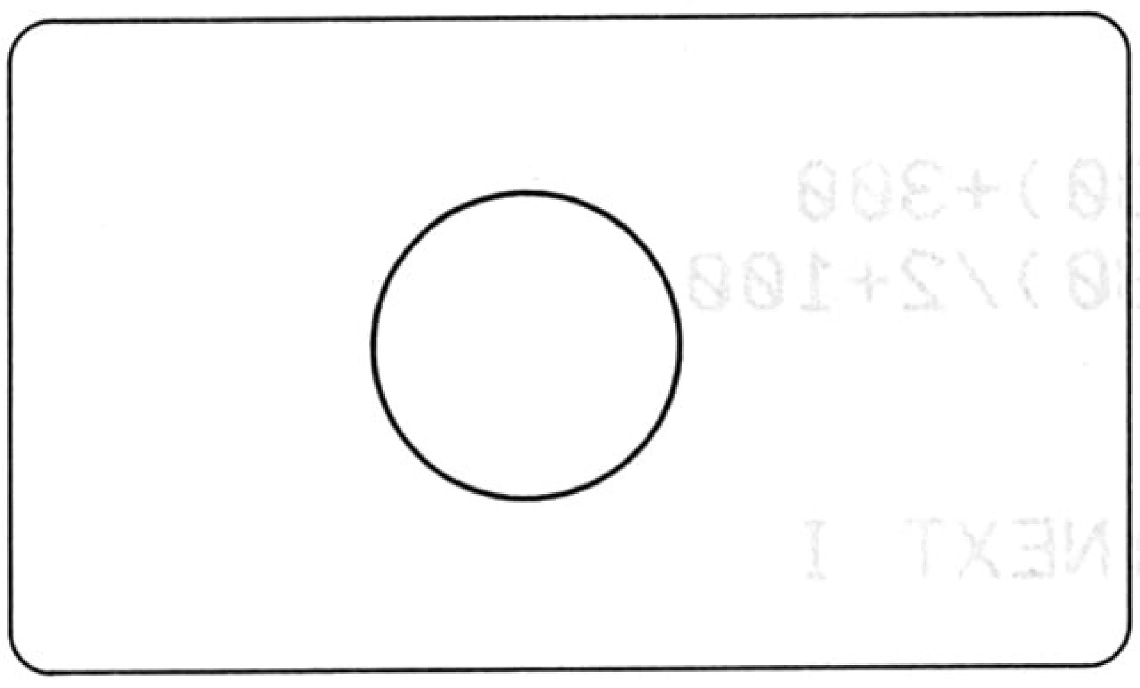
```
100 CLS 3
110 FOR J=0 TO 360
120   X=100*COS(J*3.14159/180)+300
130   Y=100*SIN(J*3.14159/180)/2+100
140   PSET (X,Y)
150 NEXT J
160 END
```

〔キーイン後の説明〕

① キーインしたプログラムは，三角関数を使って円を描くものです。

設問 42-2 プログラムを実行して下さい。

<実行結果例>



設問 42-3 ファンクションキー（f･1～f･7キー）が押されたら，色を変更するような処理を追加します。

<追加する処理>

```
10 ON KEY GOSUB *F1,*F2,*F3,*F4,*F5,*F6,*F7
20 FOR I=1 TO 7:KEY(I) ON:NEXT I
155 FOR I=1 TO 7:KEY(I) OFF:NEXT I
1000 *F1:KEY(1) OFF:COLOR ,,,1:KEY(1) ON:RETURN
1010 *F2:KEY(2) OFF:COLOR ,,,2:KEY(2) ON:RETURN
1020 *F3:KEY(3) OFF:COLOR ,,,3:KEY(3) ON:RETURN
1030 *F4:KEY(4) OFF:COLOR ,,,4:KEY(4) ON:RETURN
1040 *F5:KEY(5) OFF:COLOR ,,,5:KEY(5) ON:RETURN
1050 *F6:KEY(6) OFF:COLOR ,,,6:KEY(6) ON:RETURN
1060 *F7:KEY(7) OFF:COLOR ,,,7:KEY(7) ON:RETURN
```

〔追加後の説明〕

① 行番号10は，ファンクションキーが押されたときに実行するサブルーチンを定義しています。これによって，f･1キーが押されたときはサブルーチン＊F１の処理，f･2キーが押されたときは＊F２の処理といった具合に実行されます。

② 行番号20は，ファンクションキーによる割り込みをf･1～f･7キーまでの７個について許可します。

③ 行番号155は，プログラムの実行を終了する前にファンクションキーによる割り込みを禁止するようにします。

④ 行番号1000～1060は，f･1～f･7までのいずれかのキーが押されたときの処理（色の変更）を行っています。また，ここではさらにサブルーチンの実行中には割り込みが起こらないようにしています。

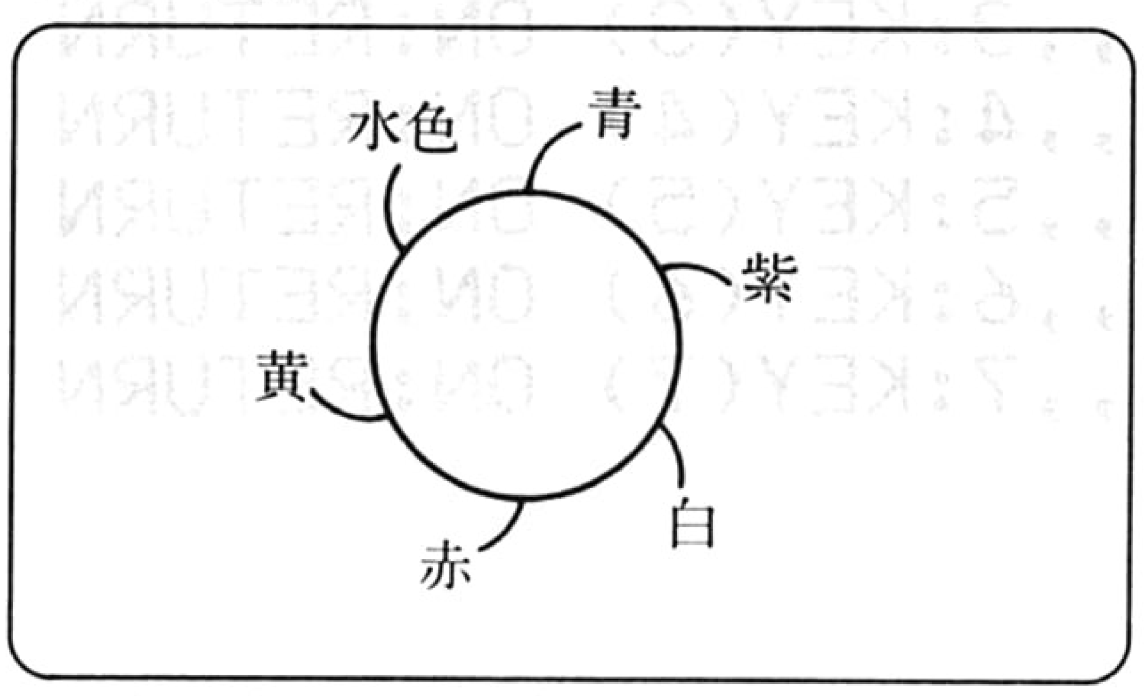
設問 42-4　プログラムのリスト画面に表示して，正しくキーインできているかどうか確認して下さい。

＜確認用リスト＞

```
10 ON KEY GOSUB *F1,*F2,*F3,*F4,*F5,*F6,*F7
20 FOR I=1 TO 7:KEY(I) ON:NEXT I
100 CLS 3
110 FOR J=0 TO 360
120   X=100*COS(J*3.14159/180)+300
130   Y=100*SIN(J*3.14159/180)/2+100
140   PSET (X,Y)
150 NEXT J
155 FOR I=1 TO 7:KEY(I) OFF:NEXT I
160 END
1000 *F1:KEY(1) OFF:COLOR ,,,1:KEY(1) ON:RETURN
1010 *F2:KEY(2) OFF:COLOR ,,,2:KEY(2) ON:RETURN
1020 *F3:KEY(3) OFF:COLOR ,,,3:KEY(3) ON:RETURN
1030 *F4:KEY(4) OFF:COLOR ,,,4:KEY(4) ON:RETURN
1040 *F5:KEY(5) OFF:COLOR ,,,5:KEY(5) ON:RETURN
1050 *F6:KEY(6) OFF:COLOR ,,,6:KEY(6) ON:RETURN
1060 *F7:KEY(7) OFF:COLOR ,,,7:KEY(7) ON:RETURN
```

設問 42-5　正しくキーインできていたら，プログラムを実行して下さい。

＜実行結果例＞



〔実行後の説明〕

① プログラムを実行すると，右横の位置から右回りに円を描いていきます。

② 円を描いていく途中で，[f･1]～[f･7]までの任意のキーを押すと割り込み処理としてフォアグラウンドカラーの変更を行い，対応するカラーコードの色で円を描きます。

【命令のまとめ】

ファンクションキーが押されたとき割り込み機能が働くように定義する命令と，割り込みを認めるか認めないかを決める命令の一般形式と書き方例を次に示します。

＜一般形式＞

ON KEY GOSUB 行番号1，行番号2，行番号3，……

・ファンクションキーを押したときに割り込み処理を行うサブルーチンを指定します。

・行番号1は，[f・1]キーを押したときに割り込み処理を行うサブルーチンの実行開始行番号を指定します。

・行番号2は，[f・2]キーを押したときに割り込み処理を行うサブルーチンの実行開始番号を指定します。

・行番号3以降も同様です。

KEY（n） ON／OFF／STOP

・ファンクションキー（n）による割り込みを認めるかどうかを，許可（ON），保留（STOP），禁止（OFF）のいずれかで指定します。

＜書き方例＞

① ON KEY GOSUB 1000
KEY（1） ON
⋮

この例は，[f・1]キーによる割り込みを認め，割り込みが起こったとき「行番号1000」のサブルーチンを実行します。

② ON KEY GOSUB ＊PCSUB，＊MCSUB
KEY(1) ON
KEY(2) ON
⋮

この例は，[f・1]キーと[f・2]キーの割り込みを認め，[f・1]キーによる割り込みが起こったとき「＊PCSUB」というラベル名のサブルーチンを実行します。また，[f・2]キーによる割り込みが起こったとき「＊MCSUB」というラベル名のサブルーチンを実行します。

4.1.2 ストップキーによる割り込み　（ON　STOP　GOSUB）

これは，プログラムを実行している最中に，ストップキーが押されたとき割り込み機能が働くように定義するものです。

ストップキーによる割り込みを定義すると，通常のストップキーとしての機能（強制停止）が失われます。したがって，不用意にこの処理を行うとプログラムの実行を停止することができなくなる場合があるので注意して下さい。

問題　43　問題42で作成したプログラムに，ストップキーが押されたら画面をクリアして最初から円を描き直す機能を追加して下さい。

【 要点の説明 】

ストップキーによる割り込み処理も，ファンクションキーの場合と同様に割り込みの時の処理を定義した後，割り込みを認めるかどうかを設定します。

【 プログラムの作成 】

設問　43-1　現在メモリーに入っているプログラムに，次の命令を追加して下さい。

<追加する命令>

```
30 ON STOP GOSUB *STP
40 STOP ON
157 STOP OFF
1070 *STP:STOP OFF:CLS 2:STOP ON:RETURN 40
```

〔追加後の説明〕

① 行番号30は，ストップキーが押されたときに実行するサブルーチンを定義しています。

② 行番号40は，ストップキーによる割り込みを許可する状態にしています。この命令が実行された後は，ストップキーを押してもプログラムの実行は止まりません。(「ON　STOP GOSUB」で定義されたサブルーチンが実行されます)

③ 行番号157は，プログラムの実行を終了する前にストップキーによる割り込みを禁止する状態にしています。ストップキーによる割り込みを行うプログラムは，終了時に必ずこの命令を実行します。

④　行番号1070は，ストップキーが押されたときの割り込み処理です。これを実行した後は，行番号40から実行が再開されます。

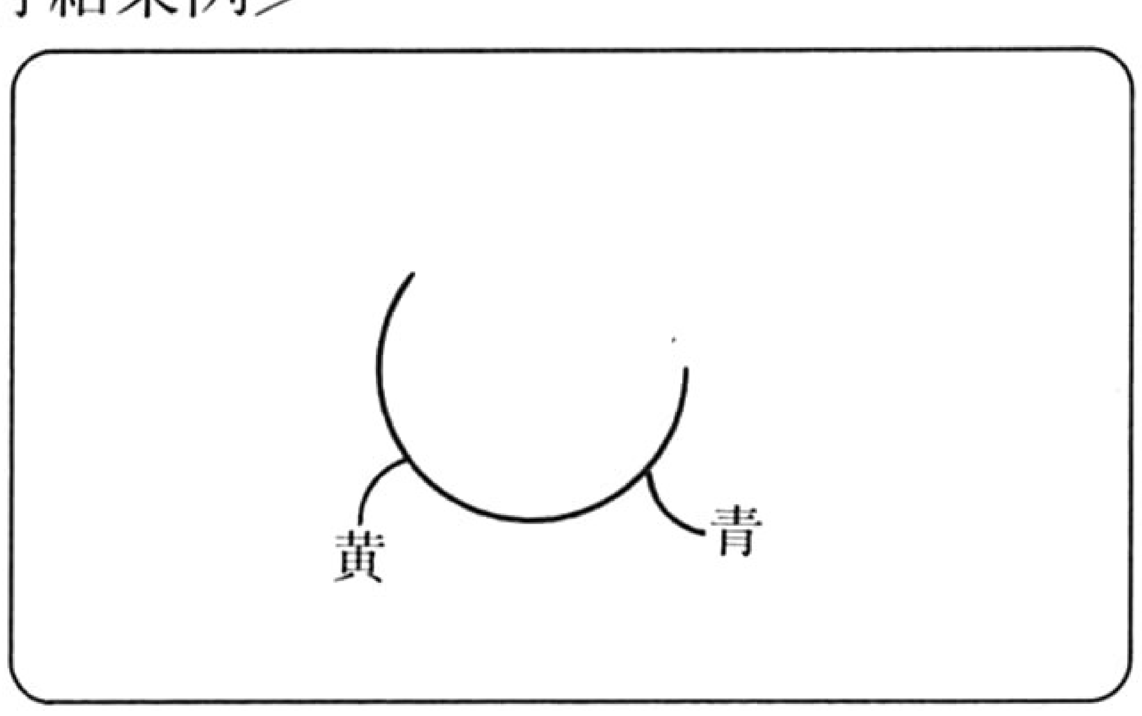
設問 43-2　プログラムのリストを画面に表示して，正しくキーインできているかどうか確認して下さい。

<確認用リスト>

```
10 ON KEY GOSUB *F1,*F2,*F3,*F4,*F5,*F6,*F7
20 FOR I=1 TO 7:KEY(I) ON:NEXT I
30 ON STOP GOSUB *STP
40 STOP ON
100 CLS 3
110 FOR J=0 TO 360
120   X=100*COS(J*3.14159/180)+300
130   Y=100*SIN(J*3.14159/180)/2+100
140   PSET (X,Y)
150 NEXT J
155 FOR I=1 TO 7:KEY(I) OFF:NEXT I
157 STOP OFF
160 END
1000 *F1:KEY(1) OFF:COLOR ,,,1:KEY(1) ON:RETURN
1010 *F2:KEY(2) OFF:COLOR ,,,2:KEY(2) ON:RETURN
1020 *F3:KEY(3) OFF:COLOR ,,,3:KEY(3) ON:RETURN
1030 *F4:KEY(4) OFF:COLOR ,,,4:KEY(4) ON:RETURN
1040 *F5:KEY(5) OFF:COLOR ,,,5:KEY(5) ON:RETURN
1050 *F6:KEY(6) OFF:COLOR ,,,6:KEY(6) ON:RETURN
1060 *F7:KEY(7) OFF:COLOR ,,,7:KEY(7) ON:RETURN
1070 *STP:STOP OFF:CLS 2:STOP ON:RETURN 40
```

設問 43-3　正しくキーインできていたら，プログラムを実行して下さい。

<実行結果例>



〔実行後の説明〕

①　ストップキーを押すと，画面がクリアされて新たに円を描き直します。

【命令のまとめ】

ストップキーが押されたとき割り込み機能が働くように定義する命令と，割り込みを認めるかどうかを設定する命令の一般形式と書き方例を次に示します。

<一般形式>

ON　STOP　GOSUB　行番号

・STOPキーを押して割り込みが起こったときに実行するサブルーチンの実行開始行番号を指定します。

STOP　ON／OFF／STOP

・STOPキーによる割り込みを認めるかどうかを，許可（ON），保留（STOP），禁止（OFF）のいずれかで指定します。

<書き方例>

① ON　STOP　GOSUB　1000
　STOP　ON
　　⋮

これは，STOPキーの割り込みを認めSTOPキーが押されたときに「行番号1000」のサブルーチンを実行します。

② ON　STOP　GOSUB　＊LOOP
　STOP　ON
　　⋮

これも①と同様に，STOPキーによる割り込みを認め，割り込みが起こったとき「＊LOOP」というラベル名のサブルーチンを実行します。

4.1.3 ヘルプキーによる割り込み　(ON HELP GOSUB)

これは，プログラムを実行している最中に，ヘルプキーが押されたとき割り込み機能が働くように定義するものです。

ヘルプキーによる割り込み処理は，ファンクションキーやストップキーの場合と違って何か処理を行うのではなく，プログラム実行の手助けとなる操作説明の表示などを行うことが多いようです。

問題　44　問題42と43で作成したプログラムに，ヘルプキーが押されたら操作説明を表示する機能を追加して下さい。

【 プログラムの作成 】

設問 44-1　現在メモリーに入っているプログラムに，次の命令を追加して下さい。

＜追加する命令＞

```
50 ON HELP GOSUB *HLP
60 HELP ON
159 HELP OFF
1080 *HLP:HELP OFF
1090 SCREEN ,2
1100 PRINT "ｺﾉﾌﾟﾛｸﾞﾗﾑﾊ ｻﾞﾋｮｳ(300,100)ﾉｲﾁﾆ ﾊﾝｹｲ<100>ﾉｴﾝｦ ｴｶﾞｷﾏｽ
1110 PRINT
1120 PRINT "ﾌﾟﾛｸﾞﾗﾑﾉｼﾞｯｺｳﾁｭｳﾆ ﾂｷﾞﾉｷｰｦｵｽﾄ ﾐｷﾞﾉ ｼｮﾘｦ ｵｺﾅｲﾏｽ
1130 PRINT
1140 PRINT "        f･1   ---   ｱｵｲﾛﾆﾍﾝｺｳｼﾏｽ
1150 PRINT
1160 PRINT "        f･2   ---   ｱｶｲﾛﾆﾍﾝｺｳｼﾏｽ
1170 PRINT
1180 PRINT "        f･3   ---   ﾑﾗｻｷｲﾛﾆﾍﾝｺｳｼﾏｽ
1190 PRINT
1200 PRINT "        f･4   ---   ﾐﾄﾞﾘｲﾛﾆﾍﾝｺｳｼﾏｽ
1210 PRINT
1220 PRINT "        f･5   ---   ﾐｽﾞｲﾛﾆﾍﾝｺｳｼﾏｽ
1230 PRINT
1240 PRINT "        f･6   ---   ｷｲﾛﾆﾍﾝｺｳｼﾏｽ
1250 PRINT
1260 PRINT "        f･7   ---   ｼﾛｲﾛﾆﾍﾝｺｳｼﾏｽ
1270 PRINT
1280 PRINT "        STOP  ---   ｻｲｼﾞｯｺｳｦ ｵｺﾅｲﾏｽ
1290 PRINT
1300 PRINT "        HELP  ---   ｿｳｻｾﾂﾒｲｦ ﾋｮｳｼﾞｼﾏｽ
1310 A$=INPUT$(1)
1320 HELP ON:CLS:SCREEN ,0:RETURN
```

〔追加後の説明〕

① 行番号50は，ヘルプキーが押されたときに実行するサブルーチンを定義しています。

② 行番号60は，ヘルプキーによる割り込みを許可する状態にしています。

③ 行番号159は，プログラムの実行を終了する前にヘルプキーによる割り込みを禁止する状態にしています。

④ 行番号1080～1310は，ヘルプキーが押されたときに実行する処理です。

⑤ 行番号1090は，グラフィック画面をマスクして，描いている途中の円を見えないようにしています。

⑥ 行番号1100～1300は，プログラムの操作説明の表示です。

⑦ 行番号1310は，英字または数字などのキーが押されたときに表示を終了して割り込み処理を終わるようにしています。

⑧ 行番号1320は，画面に表示されているメッセージをクリアしてグラフィック画面のマスクを取って円を表示した後，元の処理に戻ります。

設問 44-2 プログラムのリストをプリンタに印字して，正しくキーインできているかどうか確認して下さい。

＜確認用リスト＞

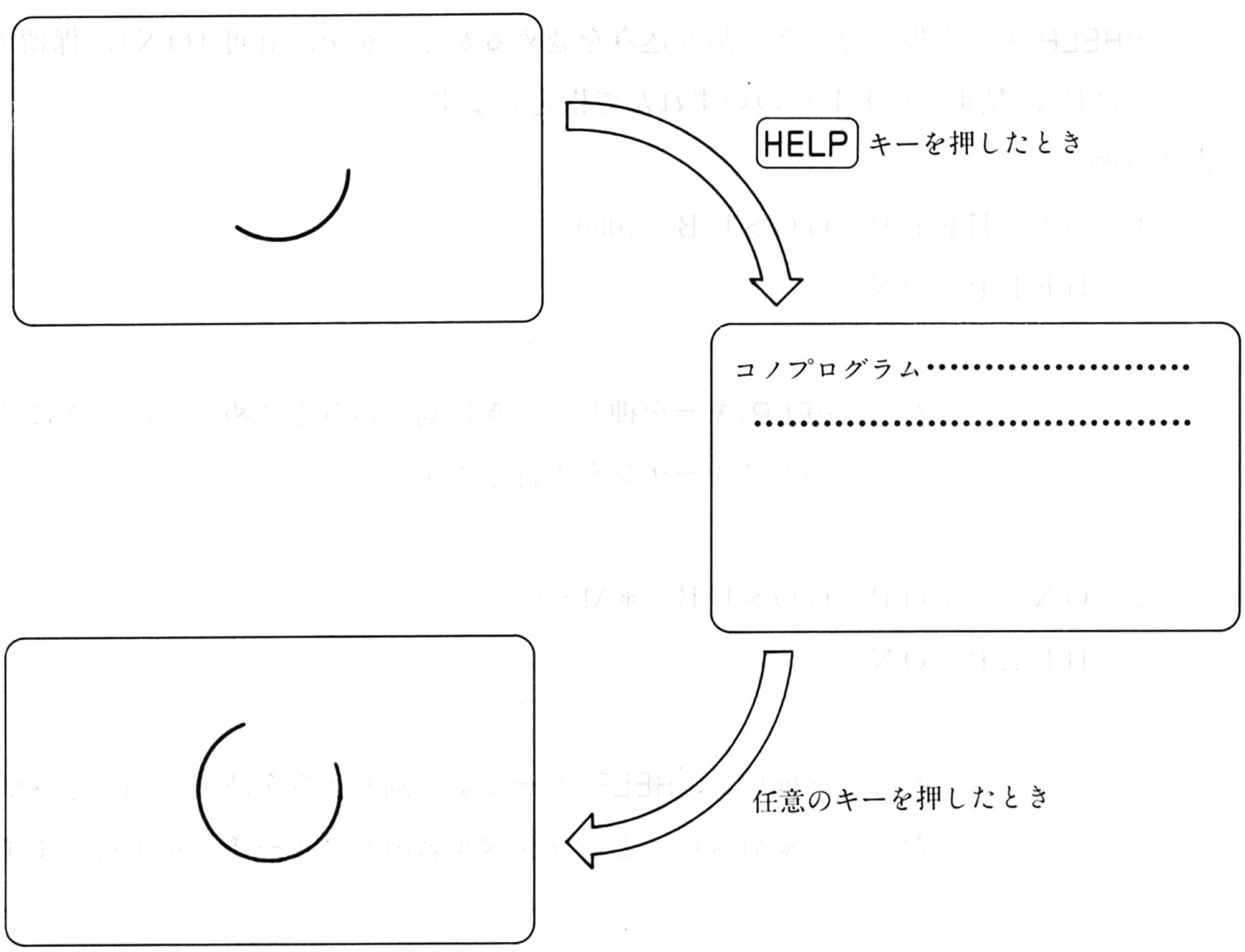
```
10 ON KEY GOSUB *F1,*F2,*F3,*F4,*F5,*F6,*F7
20 FOR I=1 TO 7:KEY(I) ON:NEXT I
30 ON STOP GOSUB *STP
40 STOP ON
50 ON HELP GOSUB *HLP
60 HELP ON
100 CLS 3
110 FOR J=0 TO 360
120   X=100*COS(J*3.14159/180)+300
130   Y=100*SIN(J*3.14159/180)/2+100
140   PSET (X,Y)
150 NEXT J
155 FOR I=1 TO 7:KEY(I) OFF:NEXT I
157 STOP OFF
159 HELP OFF
160 END
1000 *F1:KEY(1) OFF:COLOR ,,,1:KEY(1) ON:RETURN
1010 *F2:KEY(2) OFF:COLOR ,,,2:KEY(2) ON:RETURN
1020 *F3:KEY(3) OFF:COLOR ,,,3:KEY(3) ON:RETURN
1030 *F4:KEY(4) OFF:COLOR ,,,4:KEY(4) ON:RETURN
1040 *F5:KEY(5) OFF:COLOR ,,,5:KEY(5) ON:RETURN
1050 *F6:KEY(6) OFF:COLOR ,,,6:KEY(6) ON:RETURN
1060 *F7:KEY(7) OFF:COLOR ,,,7:KEY(7) ON:RETURN
1070 *STP:STOP OFF:CLS 2:STOP ON:RETURN 40
1080 *HLP:HELP OFF
1090 SCREEN ,2
1100 PRINT "ｺﾉﾌﾟﾛｸﾞﾗﾑﾊ ｻﾞﾋｮｳ(300,100)ﾉｲﾁﾆ ﾊﾝｹｲ<100>ﾉｴﾝｦ ｴｶﾞｷﾏｽ
```

（つづく）

```
1110 PRINT
1120 PRINT "ﾌﾟﾛｸﾞﾗﾑﾉｼﾞｯｺｳﾁｭｳﾆ ﾂｷﾞﾉｷｰｦｵｽﾄ ﾐｷﾞﾉ ｼｮﾘｦ ｵｺﾅｲﾏｽ
1130 PRINT
1140 PRINT "        f･1   ---   ｱｵｲﾛﾆﾍﾝｺｳｼﾏｽ
1150 PRINT
1160 PRINT "        f･2   ---   ｱｶｲﾛﾆﾍﾝｺｳｼﾏｽ
1170 PRINT
1180 PRINT "        f･3   ---   ﾑﾗｻｷｲﾛﾆﾍﾝｺｳｼﾏｽ
1190 PRINT
1200 PRINT "        f･4   ---   ﾐﾄﾞﾘｲﾛﾆﾍﾝｺｳｼﾏｽ
1210 PRINT
1220 PRINT "        f･5   ---   ﾐｽﾞｲﾛﾆﾍﾝｺｳｼﾏｽ
1230 PRINT
1240 PRINT "        f･6   ---   ｷｲﾛﾆﾍﾝｺｳｼﾏｽ
1250 PRINT
1260 PRINT "        f･7   ---   ｼﾛｲﾛﾆﾍﾝｺｳｼﾏｽ
1270 PRINT
1280 PRINT "        STOP ---   ｻｲｼﾞｯｺｳｦ ｵｺﾅｲﾏｽ
1290 PRINT
1300 PRINT "        HELP ---   ｿｳｻｾﾂﾒｲｦ ﾋｮｳｼﾞｼﾏｽ
1310 A$=INPUT$(1)
1320 HELP ON:CLS:SCREEN ,0:RETURN
```

設問 **44-3　正しくキーインできていたら，プログラムを実行して下さい。**

<実行結果例>



〔実行後の説明〕

① プログラム実行中にHELPキーを押すと割り込みがかかり，操作説明のメッセージが表示されます。その後，任意のキーを押すことによって割り込み処理が終了して，画面が元に戻って実行が再開されます。

設問 44-3　このプログラムをドライブ 2 のフロッピーディスクに「MONDAI.44」という名前で保存して下さい。

<実行結果例>

```
save "2:MONDAI.44"
Ok
```

【命令のまとめ】

ヘルプキーが押されたとき割り込み機能が働くように定義する命令と，割り込みを認めるかどうかを設定する命令の一般形式と書き方例を次に示します。

<一般形式>

ON HELP GOSUB 行番号

・HELPキーを押して割り込みが起こったときに実行するサブルーチンの実行開始行番号を指定します。

HELP ON／OFF／STOP

・HELPキーを押したときに割り込みを認めるかどうかを，許可（ON），保留（STOP），禁止（OFF）のいずれかで指定します。

<書き方例>

① ON HELP GOSUB 1000
　HELP ON
　　⋮

これは，HELPキーを押したときに割り込みを認め，そのときに「行番号1000」のサブルーチンを実行します。

② ON STOP GOSUB ＊MSG
　HELP ON
　　⋮

これも①と同様に，HELPキーによる割り込みを認め，割り込みが起こったとき「＊MSG」というラベル名のサブルーチンを実行します。

4.1.4 その他の割り込み

これまでキーボードからの割り込みについて，具体例を掲げながら説明してきました。このほかの割り込みもキーボードの場合と同じように定義して実行します。ここでは，今までに説明しなかった割り込み処理の命令について次に説明します。

【命令のまとめ】

(1) ON COM GOSUB ～， COM ON／OFF／STOP

RS-232C通信回線から割り込み機能が働くように定義する命令と，割り込みを認めるかどうかを設定する命令の一般形式と書き方例を次に示します。

＜一般形式＞

ON COM GOSUB 行番号

- 通信回線からの割り込みが起こったときに実行するサブルーチンの実行開始行番号を指定します。
- PC-8801mkIIには，RS-232Cポートが標準で一つ装備されています。
- 行番号は，標準装備されているRS-232Cポートに割り込みが起こったときに実行するサブルーチンの実行開始行番号を指定します。

COM ON／OFF／STOP

- RS-232C通信回線からの割り込みを認めるかどうかを，許可（ON），保留（STOP），禁止（OFF）のいずれかで指定します。

＜書き方例＞

①
```
ON COM GOSUB 1000
COM ON
   :
```

これは，本体のRS-232Cポートに割り込みが起こったときに行番号1000のサブルーチンを実行するようにします。

(2) ON　PEN　GOSUB　~，PEN　ON／OFF／STOP

ライトペンによって割り込み機能が働くように定義する命令と，割り込みを認めるかどうかを設定する命令の一般形式と書き方例を次に示します。

<一般形式>

ON　PEN　GOSUB　行番号

・ライトペンが押されて割り込みが起こったときに実行するサブルーチンの実行開始行番号を指定します。

PEN　ON／OFF／STOP

・ライトペンが押されたときに割り込みを認めるかどうかを，許可（ON），保留（STOP），禁止（OFF）のいずれかで指定します。

<書き方例>

① ON　PEN　GOSUB　1000
PEN　ON
⋮

これは，ライトペンが押されたときに実行する割り込み処理を行番号1000のサブルーチンとしておいて，ライトペンが押されたときにそのサブルーチンを実行します。

(3) ON　TIME＄　GOSUB~，TIME＄　ON／OFF／STOP

ある一定の時刻に割り込み機能が働くように定義する命令と，割り込みを認めるかどうかを設定する命令の一般形式と書き方例を次に示します。

<一般形式>

ON　TIME＄＝"時：分：秒"　GOSUB　行番号

・"時：分：秒"で指定した時刻に実行するサブルーチンの実行開始行番号を指定します。

TIMES＄　ON／OFF／STOP

・指定した時刻になったときに割り込みを認めるかどうかを，許可（ON），保留（STOP），禁止（OFF）のいずれかで指定します。

<書き方例>

① ON　TIME＄＝"10：00：00"　GOSUB　1000
TIME＄　ON
⋮

これは，10時00分00秒に行番号1000のサブルーチンを実行します。

4.2 エラー処理 (ON ERROR GOTO)

一般に割り込み処理というと，前項で説明したように外部からの要求によるものですが，PC-8801mk IIでは，プログラム実行中にエラーが発生した場合にも，割り込み処理と同じようにあらかじめ設定しておいた処理を実行することができます。これは，「エラートラップ機能」と呼ばれ，プログラムの実行中にエラーが発生したかどうかをいつも監視している機能です。

エラートラップ機能が他の割り込み処理と違う点は，プログラムのある時点で割り込みを保留したり禁止したりすることができない点です。したがって，エラーの監視はプログラムの実行中常に行われています。

> **問題 45** 問題39で作成したプログラムは，タイリングを行う際の入力の仕方によってエラーが発生します。そこで，このプログラムにエラートラップ機能を追加して，エラーを回避するようにして下さい。

【 プログラムの追加 】

設問 45-1 「@load.n88」というプログラムを実行後，次のようにドライブ 2 のフロッピーディスクから「MONDAI.39」という名前のプログラムをメモリーにロードして下さい。

<実行結果例>

```
load "2:MONDAI.39"
Ok
```

(注)「@load.n88」については，「2.6 タートルの動かし方」を参照

設問 45-2 いまメモリーにロードしたプログラムのリストをプリンタに印字して下さい。

<印字例>

```
100 SCREEN 0,0:CMD TURTLE "HT":CMD CLS 3
102 DIM G%(3002)
110 POINT (50,25)
120 CMD TURTLE "ST":TSW=1:PSW=1
130 CLS:N$=INPUT$(1):N%=VAL(N$)
132 IF N$>"9" THEN GOSUB *GSC:GOTO 130
```

(つづく)

```
140 ON N% GOSUB *N1,*N2,*N3,*N4,*N5,*N6,*N7,*N8,*N9
150 GOTO 130
160 *N1:CMD TURTLE "HD225 FD1":RETURN
170 *N2:CMD TURTLE "HD180 FD1":RETURN
180 *N3:CMD TURTLE "HD135 FD1":RETURN
190 *N4:CMD TURTLE "HD270 FD1":RETURN
200 *N5:CMD TURTLE "FD1":RETURN
210 *N6:CMD TURTLE "HD90 FD1":RETURN
220 *N7:CMD TURTLE "HD315 FD1":RETURN
230 *N8:CMD TURTLE "HD0 FD1":RETURN
240 *N9:CMD TURTLE "HD45 FD1":RETURN
250 *GSC:' ──── ｸﾞﾗﾌｨｯｸ ｻﾌﾞｺﾏﾝﾄﾞ ────
251  IF N$><"E" THEN 260
252  FOR X=0 TO 480 STEP 160
253    FOR Y=0 TO 100 STEP 100
254      GET@(X,Y)-STEP(159,99),G%
255      BSAVE "2:ZUKEI"+STR$(X+Y),VARPTR(G%(0)),6006
256  NEXT Y,X
257  END
260  IF N$><"T" THEN 290
270     IF TSW=1 THEN T$="HT" ELSE T$="ST"
280     CMD TURTLE T$:TSW=1-TSW:RETURN
290  IF N$><"F" THEN 320
300     INPUT "ｽｽﾒﾙｶｽﾞﾊ";N
310     CMD TURTLE "FD(N)":RETURN
320  IF N$><"B" THEN 350
330     INPUT "ｺｳﾀｲｻｾﾙｶｽﾞﾊ";N
340     CMD TURTLE "BK(N)":RETURN
350  IF N$><"L" THEN 370
360     CMD TURTLE "LT1":RETURN
370  IF N$><"R" THEN 390
380     CMD TURTLE "RT1":RETURN
390  IF N$><"C" THEN 420
400     INPUT "ｶﾗｰﾊﾞﾝｺﾞｳ";N
410     CMD TURTLE "PC(N)":RETURN
420  IF N$><"P" THEN 450
430     IF PSW=1 THEN P$="PU" ELSE P$="PD"
440     CMD TURTLE P$:PSW=1-PSW:RETURN
450  IF N$><"M" THEN 480
460     INPUT "ｲﾄﾞｳｽﾙｲﾁﾊ    x,y ";X,Y
470     CMD TURTLE "MV(X),(Y)":RETURN
480  IF N$><"@" THEN RETURN
490     INPUT "ｷｮｳｶｲｾﾝ ﾉ ｲﾛ";C2
500     IF C2<0 OR C2>7 THEN 490
510     INPUT "ﾇﾘﾂﾌﾞｽ ｲﾛ";C1
520     IF C1<0 OR C1>7 THEN 540
530        PAINT STEP(0,0),C1,C2:RETURN
540     INPUT "ﾀｲﾙﾊﾟﾀｰﾝ ﾉ ﾗｲﾝｽｳ";L
550     TILE$=""
560     FOR I=1 TO L
570       INPUT "ﾌﾟﾚｰﾝ1 ﾉ ﾊﾟﾀｰﾝ (xx) ";A$:A$=CHR$(VAL("&H"+A$))
580       INPUT "ﾌﾟﾚｰﾝ2 ﾉ ﾊﾟﾀｰﾝ (xx) ";B$:B$=CHR$(VAL("&H"+B$))
590       INPUT "ﾌﾟﾚｰﾝ3 ﾉ ﾊﾟﾀｰﾝ (xx) ";C$:C$=CHR$(VAL("&H"+C$))
600       TILE$=TILE$+A$+B$+C$
610     NEXT I
620     PAINT STEP(0,0),TILE$,C2:RETURN
```

〔ロードしたプログラムの説明〕

① タイリングを行うときに発生すると考えられるエラーは,「Illegal function call」と「Out of memory」の二つのエラーがPAINT命令実行時に発生することです。

② 「Illegal function call」のエラーが発生する原因としては, ペイントを行うときにタートルが画面の表示内にいない場合(ペイントを行う位置がビューポートから外れている場合)と, 行番号570〜590のタイリングパターンの入力でおかしな値を入力した場合に発生します。

③ 「Out of memory」は, 二度塗りを行うときなどに発生します。

④ いずれのエラーも, PAINT命令実行時に発生します。

設問 45-3 エラートラップ処理を行うために次の命令を追加して下さい。

〈追加する命令〉

```
10 ON ERROR GOTO *ER
1000 *ER
1010   IF ERL><530 AND ERL><620 THEN 1120
1020   IF ERR><5 THEN 1070
1030      PRINT "タートルヲヒョウジ゛シテ イチヲカクニンシタアト"
1040      PRINT "        モウイチト゛ヤリナオシテクタ゛サイ"
1050      A$=INPUT$(1)
1060      RESUME NEXT
1070   IF ERR><7 THEN 1120
1080      PRINT "イチト゛クロニヌリカエテカラ シ゛ッコウシテクタ゛サイ"
1090      PRINT "  ソレテ゛モタ゛メナハ゛アイハ タイルハ゜ターンスウヲヘラシテヤリナオシテクタ゛サイ"
1100      A$=INPUT$(1)
1110      RESUME NEXT
1120   ON ERROR GOTO 0
```

〔追加後の説明〕

① 行番号10は, プログラム実行中にエラーが発生した場合に「*ER」というラベル名ののサブルーチンを実行するようにしています。

② 行番号1010は, ERLという関数を使ってエラーが発生した行を調べ, PAINT命令を実行する530行と620行でない場合は, 行番号1120の処理を実行するようにします。

③ 行番号1020は, ERRという関数を使って発生したエラーの種類を調べ,「Illegal function call」のエラーでない場合は, 行番号1070以降の処理を実行するようにします。

④ 行番号1030と行番号1040は,「Illegal function call」のエラー発生時に行う処理です。

⑤ 行番号1060の「RESUME NEXT」命令は, エラーが発生した命令の次の命令からプログラムの実行が行われるようにするものです。なお, INPUT$関数を使って何らかのキーが押されるまで次の処理(画面クリア)を行わないようにしています。

⑥ 行番号1070〜1110は,「Out of memory」のエラーが発生したときに行う処理です。

⑦ 行番号1120は，発生したエラーが予期していないものであったとき，通常のBASICのエラーを表示して実行を停止するようにします。

ここで予期していたエラーは，「Illegal function call」と「Out of memory」の二つです。この二つのエラーに関しては，発生した場所が530行と620行であればエラーの原因と処置もはっきりしています。ですからこの二つエラーに対しては，何らかの処理を行うことができますが，これら以外のエラーが発生した場合は，原因がわかっていないので対応する処置を行うことはできません。

設問 45-4 プログラムのリストをプリンタに印字して，正しくキーインできているかどうか確認して下さい。

＜確認用リスト＞

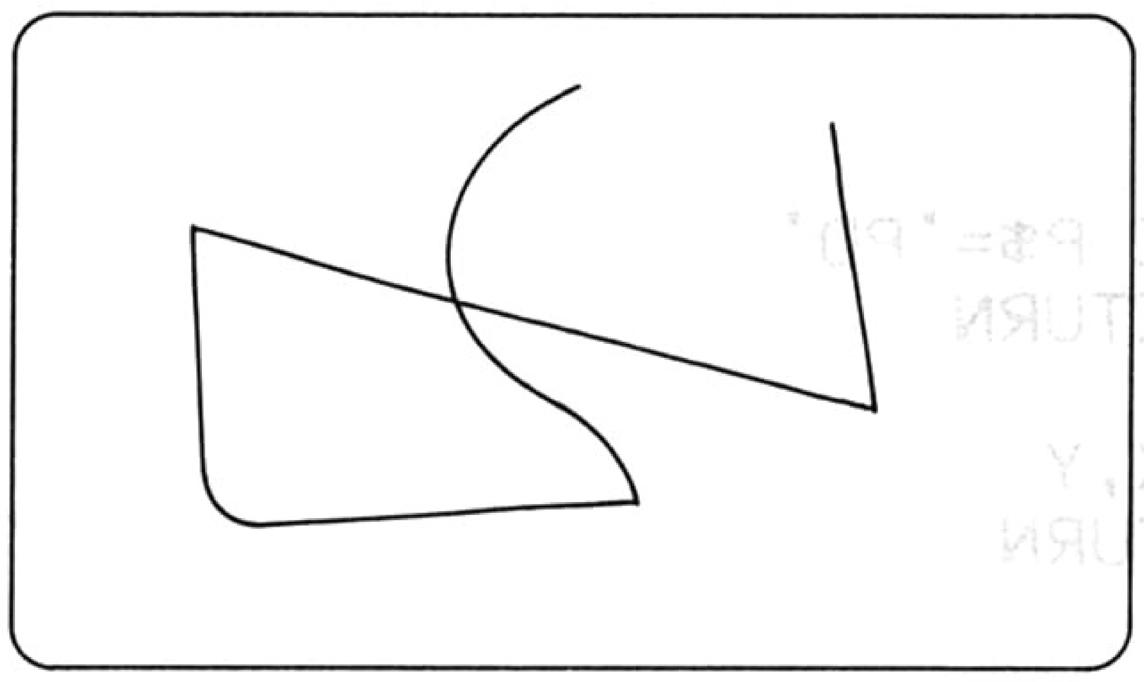
```
10 ON ERROR GOTO *ER
100 SCREEN 0,0:CMD TURTLE "HT":CMD CLS 3
102 DIM G%(3002)
110 POINT (50,25)
120 CMD TURTLE "ST":TSW=1:PSW=1
130 CLS:N$=INPUT$(1):N%=VAL(N$)
132 IF N$>"9" THEN GOSUB *GSC:GOTO 130
140 ON N% GOSUB *N1,*N2,*N3,*N4,*N5,*N6,*N7,*N8,*N9
150 GOTO 130
160 *N1:CMD TURTLE "HD225 FD1":RETURN
170 *N2:CMD TURTLE "HD180 FD1":RETURN
180 *N3:CMD TURTLE "HD135 FD1":RETURN
190 *N4:CMD TURTLE "HD270 FD1":RETURN
200 *N5:CMD TURTLE "FD1":RETURN
210 *N6:CMD TURTLE "HD90 FD1":RETURN
220 *N7:CMD TURTLE "HD315 FD1":RETURN
230 *N8:CMD TURTLE "HD0 FD1":RETURN
240 *N9:CMD TURTLE "HD45 FD1":RETURN
250 *GSC:' ------ ｸﾞﾗﾌｨｯｸ ｻﾌﾞｺﾏﾝﾄﾞ ------
251  IF N$><"E" THEN 260
252  FOR X=0 TO 480 STEP 160
253    FOR Y=0 TO 100 STEP 100
254      GET@(X,Y)-STEP(159,99),G%
255      BSAVE "2:ZUKEI"+STR$(X+Y),VARPTR(G%(0)),6006
256  NEXT Y,X
257  END
260  IF N$><"T" THEN 290
270     IF TSW=1 THEN T$="HT" ELSE T$="ST"
280     CMD TURTLE T$:TSW=1-TSW:RETURN
290  IF N$><"F" THEN 320
300     INPUT "ｽｽﾒﾙｶｽﾞﾊ";N
310     CMD TURTLE "FD(N)":RETURN
320  IF N$><"B" THEN 350
330     INPUT "ｺｳﾀｲｻｾﾙｶｽﾞﾊ";N
340     CMD TURTLE "BK(N)":RETURN
350  IF N$><"L" THEN 370
```

（つづく）

```
360      CMD TURTLE "LT1":RETURN
370   IF N$><"R" THEN 390
380      CMD TURTLE "RT1":RETURN
390   IF N$><"C" THEN 420
400      INPUT "ｶﾗｰﾊﾞﾝｺﾞｳ";N
410      CMD TURTLE "PC(N)":RETURN
420   IF N$><"P" THEN 450
430      IF PSW=1 THEN P$="PU" ELSE P$="PD"
440      CMD TURTLE P$:PSW=1-PSW:RETURN
450   IF N$><"M" THEN 480
460      INPUT "ｲﾄﾞｳｽﾙｲﾁﾊ    x,y  ";X,Y
470      CMD TURTLE "MV(X),(Y)":RETURN
480   IF N$><"@" THEN RETURN
490      INPUT "ｷｮｳｶｲｾﾝ ﾉ ｲﾛ";C2
500      IF C2<0 OR C2>7 THEN 490
510      INPUT "ﾇﾘﾂﾌﾞｽ ｲﾛ";C1
520      IF C1<0 OR C1>7 THEN 540
530         PAINT STEP(0,0),C1,C2:RETURN
540      INPUT "ﾀｲﾙﾊﾟﾀｰﾝ ﾉ ﾗｲﾝｽｳ";L
550      TILE$=""
560      FOR I=1 TO L
570        INPUT "ﾌﾟﾚｰﾝ1 ﾉ ﾊﾟﾀｰﾝ (xx)  ";A$:A$=CHR$(VAL("&H"+A$))
580        INPUT "ﾌﾟﾚｰﾝ2 ﾉ ﾊﾟﾀｰﾝ (xx)  ";B$:B$=CHR$(VAL("&H"+B$))
590        INPUT "ﾌﾟﾚｰﾝ3 ﾉ ﾊﾟﾀｰﾝ (xx)  ";C$:C$=CHR$(VAL("&H"+C$))
600        TILE$=TILE$+A$+B$+C$
610      NEXT I
620      PAINT STEP(0,0),TILE$,C2:RETURN
1000 *ER
1010  IF ERL><530 AND ERL><620 THEN 1120
1020  IF ERR><5 THEN 1070
1030      PRINT "ﾀｰﾄﾙｦﾋｮｳｼﾞｼﾃ ｲﾁｦｶｸﾆﾝｼﾀｱﾄ"
1040      PRINT "        ﾓｳｲﾁﾄﾞﾔﾘﾅｵｼﾃｸﾀﾞｻｲ"
1050      A$=INPUT$(1)
1060      RESUME NEXT
1070  IF ERR><7 THEN 1090
1080      PRINT "ｲﾁﾄﾞｸﾛﾆﾇﾘｶｴﾃｶﾗ ｼﾞｯｺｳｼﾃｸﾀﾞｻｲ"
1090      PRINT "  ｿﾚﾃﾞﾓﾀﾞﾒﾅﾊﾞｱｲﾊ ﾀｲﾙﾊﾟﾀｰﾝｽｳｦﾍﾗｼﾃﾔﾘﾅｵｼﾃｸﾀﾞｻｲ"
1100      A$=INPUT$(1)
1110      RESUME NEXT
1120  ON ERROR GOTO 0
```

設問 45-5　正しくキーインできていたら，プログラムを実行して下さい。

＜実行結果例＞



〔実行後の説明〕

① タイリングを行うときにエラーが発生しても，プログラムの実行は中断されずにエラー処理としてメッセージが表示された後，何らかのキーを押すことで実行を続行することができます。

設問 45-6　このプログラムをドライブ 2 のフロッピーディスクへ「MONDAI.45」という名前で保存して下さい。

＜実行結果例＞

```
save "2:MONDAI.45"
Ok
```

【命令のまとめ】

エラートラップ機能を利用するために必要な命令と関数について，一般形式と書き方例を次に示します。

＜一般形式＞

ON　ERROR　GOTO　行番号

・エラートラップ機能を働かせたり解除したりします。

・行番号は，エラーが発生したときに実行させるエラー処理ルーチンの開始番号を指定します。

・行番号として「0」を指定すると，エラートラップ機能が解除されます。したがって，それ以降にエラーが発生するとエラーの種類に関係なくプログラムは停止してしまいます。そのため，エラー処理ルーチンの中で「ON　ERROR　GOTO　0」を実行すると，エラー発生の状態なので，すぐにプログラムは停止してしまいます。

RESUME　NEXT

・発生したエラーを解除します。

・エラー処理ルーチンの出口に使います。

・「NEXT」を指定したときは，エラーが発生した命令の次に書かれている命令の実行に戻ります。

・「NEXT」を省略したときは，エラーが発生した命令の実行に戻ります。

・「NEXT」の代わりに行番号を指定すると，その指定した行番号で書かれている命令の実行に戻ります。

ERR

・エラーが発生したときに，そのエラーの種類を調べるための関数です。

ERL

・エラーが発生したときに，そのエラーがどの行番号の命令で起こったかを調べるための関数です。

＜書き方例＞

```
10  ON  ERROR  GOTO  500
20  FOR  J=1  TO  5
30      INPUT "X=", X%
40      Y%=X%
50      A%=10000*X%
60      B%=10000/Y%
70      PRINT "10000*X="; A%:PRINT "10000/X="; B%
80  NEXT  J
90  END
500   IF ERR=6  AND ERL=50  THES X%= 3 :RESUME
510   IF ERR=11  AND ERL=60  THEN B%=30000:RESUME NEXT
520   ON ERROR GOTO  0
```

練習問題

問題 1　次のような項目を持つランダムファイルがあります。下の問に答えて下さい。

1件目				2件目			9件目	
商品コード	商品名	単価	在庫数	商品コード		～	単価	在庫数
SCD$(1)	SNM$(1)	STK$(1)	SZK$(1)	SCD$(2)			STK$(9)	SZK$(9)
2桁	20桁	4桁	2桁					

・1件分のデータは，28桁です。

・1レコード内に9件のデータを記録するものとします。

〈問1〉　このファイルのフィールド文を定義して下さい。

〈問2〉　100件分の商品データが登録できるようなランダムファイルをドライブ2のフロッピーディスクに「ZAIKO」という名前で作成して下さい。

〈問3〉　データをキーボードから入力して問2で作成したファイルへ登録するプログラムを，次の条件で作成して下さい。

（条件）

・データとして入力する項目は，次のとおりとします。

商品コード	「1」～「100」までの数字
商品名	20桁以内（カナ文字）
単価	6桁以内（数値）
在庫数	「32767」以下の数字

・商品コードとして「999」が入力されたら，処理を終了するようにします。

・入力した商品コードを，ランダムファイル「ZAIKO」のレコード番号として使うようにします。

〈問4〉　在庫データの更新を行うプログラムを，次の条件で作成して下さい。

（条件）

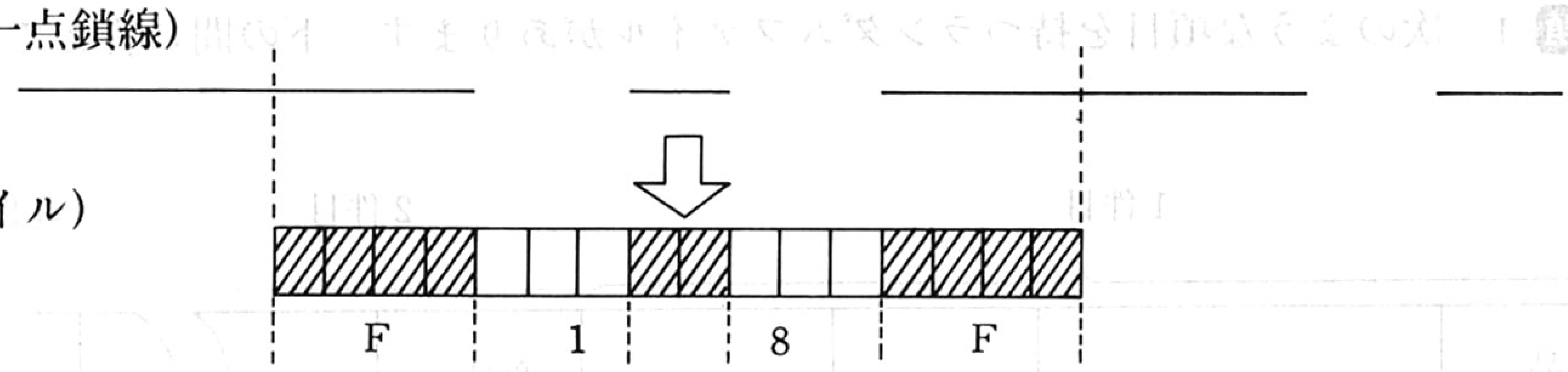
・商品コードを入力して，該当するデータの表示を行った後に入庫数を入力できるようにします。出庫の場合も「－」の値で入力できるようにします。

・該当するデータがない場合はすべての項目を入力するようにします。

問題 2　次に示すような一点鎖線を，ＬＩＮＥ命令を使って画面に表示して下さい。表示する位置は（０，100）－(639，100）です。

(表示する一点鎖線)

(ランスタイル)



問題 3　次のＬＩＮＥ命令の実行で描かれた四角形の内側を，「ＰＡＩＮＴ」命令を使って指定のタイルストリングで塗りつぶして下さい。

(実行するＬＩＮＥ命令)

ＬＩＮＥ　(200，60)－(300，100)，７，Ｂ

(タイルストリング)

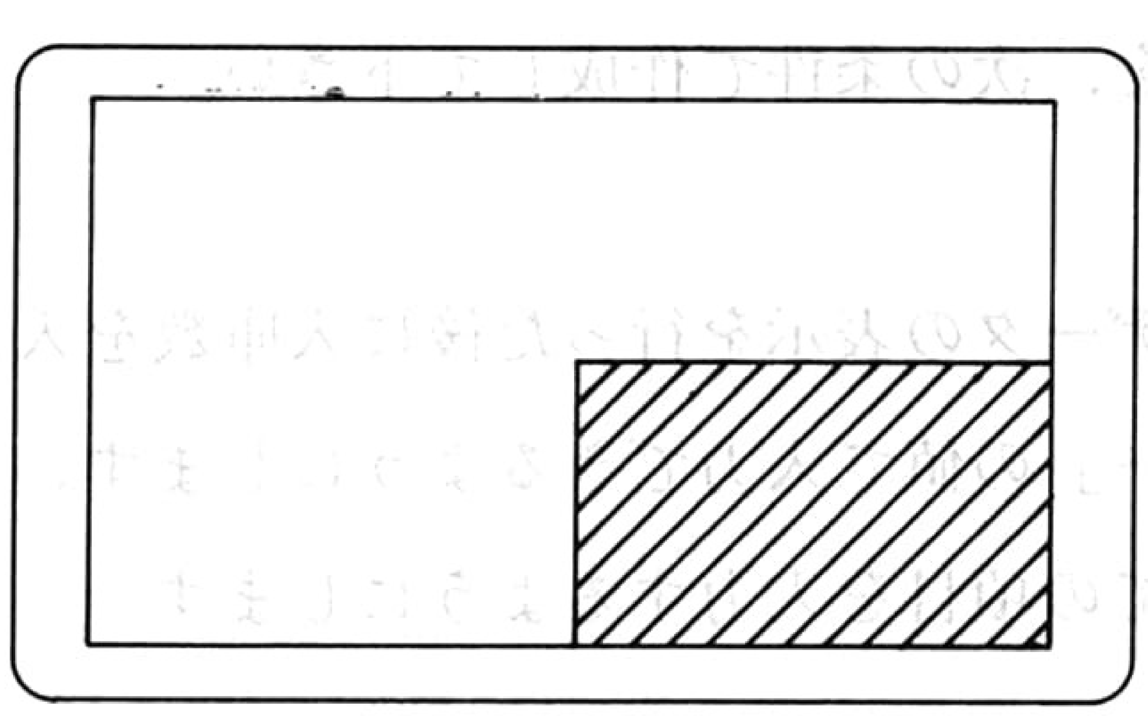
	プレーン１(青)	プレーン２(赤)	プレーン３(緑)
１行目	10	08	10
２行目	20	04	20
３行目	40	02	40
４行目	80	01	80

問題 4　次のＬＩＮＥ命令を実行したときに図のような位置に画面の1/4の大きさで四角形が表示されるように「ウィンドウ」を変えて下さい。

(ウィンドウを変えた後で実行する命令)

ＬＩＮＥ　(０，０)－(639，199)，２，ＢＦ

〈結果例〉



問題 5 次のプログラムは，キーボードから二つの数値を入力して，和，差，積，商，べき乗を求めるものです。プログラムをキーインしたのち，以下の問に答えて下さい。

(プログラムリスト)

```
10 CONSOLE 0,25,1,1
20 INPUT "ｶｽﾞ A= ",A
30 IF A=0 THEN END
40 INPUT "ｶｽﾞ B= ",B
50 PRINT "( ﾜ ) ";A+B
60 PRINT "( ｻ ) ";A-B
70 PRINT "( ｾｷ ) ";A*B
80 PRINT "( ｼｮｳ ) ";A/B
90 PRINT "(ﾍﾞｷｼﾞｮｳ) ";A^B
100 PRINT:GOTO 20
```

〈問1〉 このプログラムでは，変数Bに0を入力した場合正しい結果が得られません。そこで，〔ON ERROR GOTO〕命令を使ってエラー処理を行うように修正して下さい。

・エラー発生行　80
・エラーコード　11
・エラーメッセージ　〔Division by Zero〕
・エラー処理　"ワリザン　デキマセン"というメッセージを商の代わりに表示する。

〈問2〉 このプログラムで，もう一つ考えられるエラーはオーバーフローです。これは，べき乗を計算する時に変数Bの値が大きい場合に起こります。問1と同じように，〔ON ERROR GOTO〕を使ってエラー処理を行うようにして下さい。

・エラー発生行　90行
・エラーコード　6
・エラーメッセージ　〔Overflow〕
・エラー処理　"オーバーフロー"というメッセージをべき乗の結果の代わりに表示する。

解答例

問題 1 の解答例

〈問 1〉の解答例

```
10  FOR I=1 TO 9
20    FIELD#1,(I-1)*28    AS DM$,2  AS SCD$(I)
30    FIELD#1,(I-1)*28+2  AS DM$,20 AS SNM$(I)
40    FIELD#1,(I-1)*28+22 AS DM$,4  AS STK$(I)
50    FIELD#1,(I-1)*28+26 AS DM$,2  AS SZK$(I)
60  NEXT I
```

（解説）

1 レコード（1 セクタ）内に複数件のデータを格納するときのフィールド文の定義方法です。各データの形式が同じなので，それぞれの項目を配列変数で定義する方がわかりやすくなります。

また，同じような定義を 9 データ分行うのは大変ですから，FOR～NEXT命令を使って定義するようにします。

〈問 2〉の解答例

```
100 OPEN "2:ZAIKO" AS#1
110 FOR I=1 TO 9
120   FIELD#1,(I-1)*28    AS DM$,2  AS SCD$(I)
130   FIELD#1,(I-1)*28+2  AS DM$,20 AS SNM$(I)
140   FIELD#1,(I-1)*28+22 AS DM$,4  AS STK$(I)
150   FIELD#1,(I-1)*28+26 AS DM$,2  AS SZK$(I)
160 NEXT I
170 FOR I=1 TO 9
180   LSET SCD$(I)=MKI$(0)
190   LSET SNM$(I)=SPACE$(20)
200   LSET STK$(I)=MKS$(0)
210   LSET SZK$(I)=MKI$(0)
220 NEXT I
230 FOR I=1 TO 12
240   PUT#1,I
250 NEXT I
260 CLOSE
270 END
```

（解説）

ランダムファイルを新規に作成するには，予想されるデータ件数分の初期設定を行います。この解答例では，行番号140～190で，ランダムファイルバッファの各データ項目の初期設定を行った後，行番号200～220で書き込みを行います。

〈問３〉の解答例

```
100 *INIT:'ｼｮｷ ｼｮﾘ
110  OPEN "2:ZAIKO" AS#1
120  FOR I=1 TO 9
130    FIELD#1,(I-1)*28    AS DM$,2  AS SCD$(I)
140    FIELD#1,(I-1)*28+2  AS DM$,20 AS SNM$(I)
150    FIELD#1,(I-1)*28+22 AS DM$,4  AS STK$(I)
160    FIELD#1,(I-1)*28+26 AS DM$,2  AS SZK$(I)
170  NEXT I
180 *MAIN:'ﾒｲﾝ ﾙｰﾁﾝ
190  GOSUB *NYURYOKU.CD
200  IF CD%=999 THEN *OWARI
210  GOSUB *NYURYOKU.DATA
220  GOSUB *KAKIKOMI
230  GOTO  *MAIN
240 *OWARI:'ｼｭｳﾘｮｳ ｼｮﾘ
250  CLOSE#1
260  END
1000 *NYURYOKU.CD:'ｺｰﾄﾞ ﾆｭｳﾘｮｸ
1010  INPUT "ｼｮｳﾋﾝ ｺｰﾄﾞ";CD%
1020  IF CD%=999 THEN 1080
1030  IF CD%<1 OR CD%>100 THEN *NYURYOKU.CD
1040  RNO=CD% ¥ 9
1050  I=CD% MOD 9
1060  IF I=0 THEN I=9 ELSE RNO=RNO+1
1070  GET#1,RNO
1080  RETURN
1090 *NYURYOKU.DATA:'ﾃﾞｰﾀ ﾉ ﾆｭｳﾘｮｸ
1100  INPUT "ｼｮｳﾋﾝ ﾒｲ ";NM$
1110  INPUT "ﾀﾝｶ       ";TK
1120  INPUT "ｻﾞｲｺｽｳ    ";ZK%
1130  INPUT "  OK(y/n)";YN$
1140  IF YN$="N" OR YN$="n" THEN *NYURYOKU.DATA
1150  IF YN$<>"Y" AND YN$<>"y" THEN 1130
1160  RETURN
1170 *KAKIKOMI:'ﾃﾞｰﾀ ﾉ ｶｷｺﾐ
1180  LSET SCD$(I)=MKI$(CD%)
1190  LSET SNM$(I)=NM$
1200  LSET STK$(I)=MKS$(TK)
1210  LSET SZK$(I)=MKI$(ZK%)
1220  PUT#1,RNO
1230  RETURN
```

（解説）

・行番号150～200がプログラムの流れを制御するメインルーチンの部分です。

・行番号1000～1070は，入力された商品コードから該当するレコード番号とデータの格納位置を求めます。この中で，レコードの読み込みを行っていますが，これは１レコード内に複数のデータを格納しても，先に同じレコード内に格納されたデータを消さないようにするためです。

〈問4〉の解答例

```
100 *INIT:'ｼｮｷ ｼｮﾘ
110  OPEN "2:ZAIKO" AS#1
120  FOR I=1 TO 9
130    FIELD#1,(I-1)*28    AS DM$,2  AS SCD$(I)
140    FIELD#1,(I-1)*28+2  AS DM$,20 AS SNM$(I)
150    FIELD#1,(I-1)*28+22 AS DM$,4  AS STK$(I)
160    FIELD#1,(I-1)*28+26 AS DM$,2  AS SZK$(I)
170  NEXT I
180 *MAIN:'ﾒｲﾝ ﾙｰﾁﾝ
190  GOSUB *NYURYOKU.CD
200  IF CD%=999 THEN *OWARI
210  IF CVI(SCD$(I))=CD% THEN GOSUB *HYOUJI ELSE GOSUB *NYURYOKU.DATA
220  GOSUB *KAKIKOMI
230  GOTO  *MAIN
240 *OWARI:'ｼｭｳﾘｮｳ ｼｮﾘ
250  CLOSE#1
260  END
1000 *NYURYOKU.CD:'ｺｰﾄﾞ ﾆｭｳﾘｮｸ
1010  INPUT "ｼｮｳﾋﾝｺｰﾄﾞ";CD%
1020  IF CD%=999 THEN 1080
1030  IF CD%<1 OR CD%>100 THEN *NYURYOKU.CD
1040  RNO=CD% ¥ 9
1050  I=CD% MOD 9
1060  IF I=0 THEN I=9 ELSE RNO=RNO+1
1070  GET#1,RNO
1080  RETURN
1090 *NYURYOKU.DATA:'ﾃﾞｰﾀ ﾉ ﾆｭｳﾘｮｸ
1100  INPUT "ｼｮｳﾋﾝ ﾒｲ ";NM$
1110  INPUT "ﾀﾝｶ       ";TK
1120  INPUT "ｻﾞｲｺｽｳ    ";ZK%
1130  INPUT "   OK(y/n)";YN$
1140  IF YN$="N" OR YN$="n" THEN *NYURYOKU.DATA
1150  IF YN$<>"Y" AND YN$<>"y" THEN 1130
1160  RETURN
1170 *KAKIKOMI:'ﾃﾞｰﾀ ﾉ ｶｷｺﾐ
1180  LSET SCD$(I)=MKI$(CD%)
1190  LSET SNM$(I)=NM$
1200  LSET STK$(I)=MKS$(TK)
1210  LSET SZK$(I)=MKI$(ZK%)
1220  PUT#1,RNO
1230  RETURN
1240 *HYOUJI:'ﾃﾞｰﾀﾉ ﾋｮｳｼﾞ
1250  TK=CVS(STK$(I))
1260  ZK=CVI(SZK$(I))
1270  PRINT "ｼｮｳﾋﾝｺｰﾄﾞ";CVI(SCD$(I))
1280  PRINT "ｼｮｳﾋﾝﾒｲ  ";SNM$(I)
1290  PRINT "ﾀﾝｶ      ";CVS(STK$(I))
1300  PRINT "ｻﾞｲｺｽｳ   ";CVI(SZK$(I))
1310  PRINT
1320  INPUT "ﾆｭｳｺｽｳ   ";NK%
1330  ZK%=CVI(SZK$(I))
1340  ZK%=ZK%+NK%
1350  PRINT "ｻﾞｲｺｽｳ   ";ZK%
1360  INPUT "   OK(y/n)";YN$
1370  IF YN$="N" OR YN$="n" THEN 1310
1380  IF YN$<>"Y" AND YN$<>"y" THEN 1360
1390  LSET SZK$=MKI$(ZK%)
1400  RETURN
```

（解説）

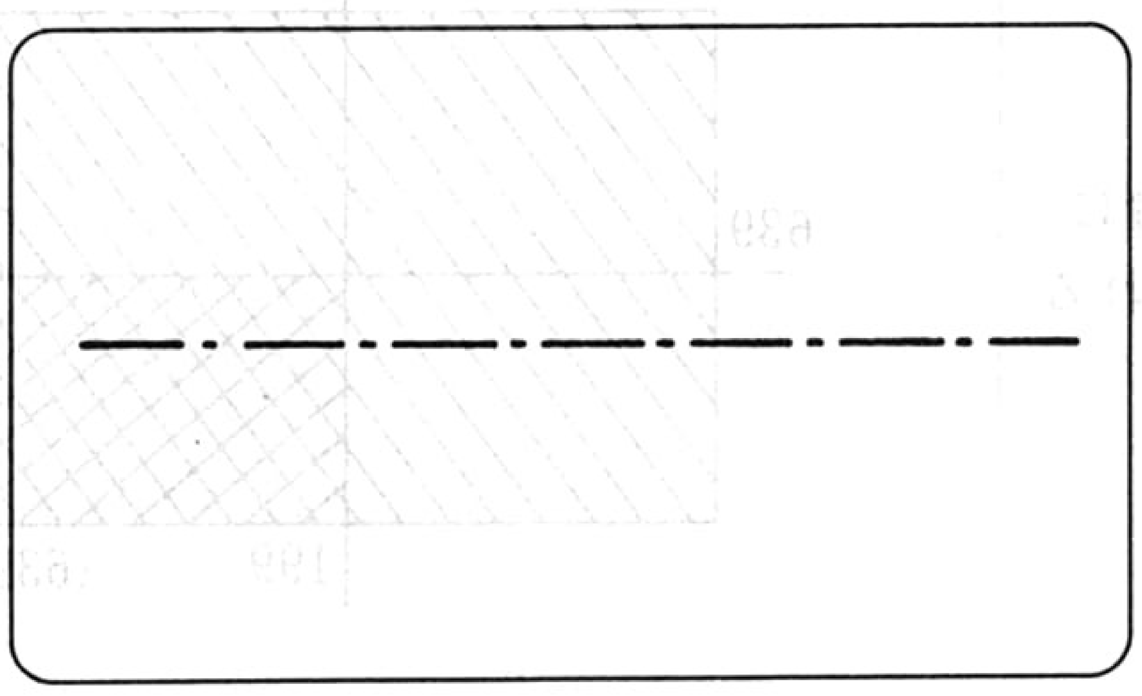
入力した商品コードに該当するデータが存在する場合は，内容を表示した後入庫数を入力するようにし，それ以外の場合はすべての項目を入力するように，〈問３〉で作成したプログラムを修正します。

ここでのランダムファイルは，商品コードに識別コードの機能を持たせて利用しています。（行番号180）

問題 2 の解答例

```
10 LINE (0,100)-(639,100),,,&HF18F
```

（実行結果例）



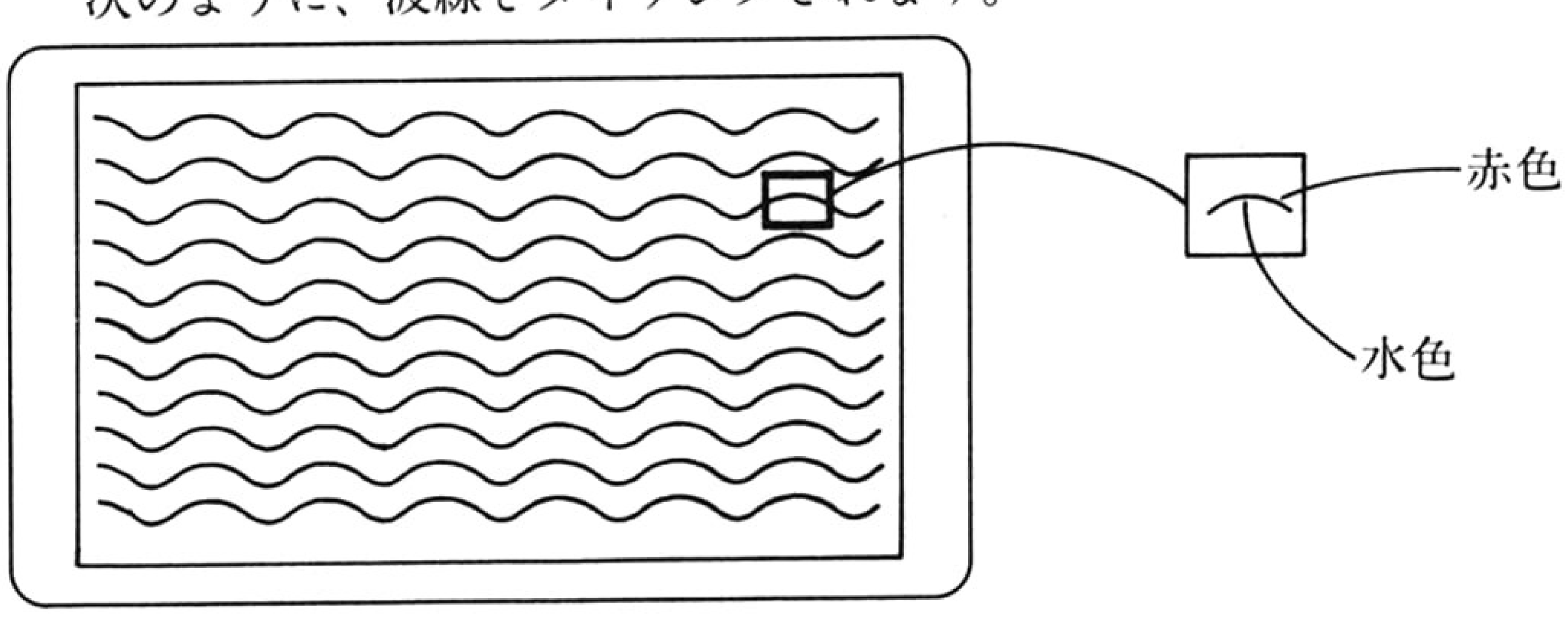
（解説）

「ＬＩＮＥ」命令を使って，このようにいろいろな線を表示できます。これには，表示しようとする線が，どういうパターンで繰り返されているかを16個のマス目で表し，それを16進数で表現して行います。

問題 3 の解答例

```
10 LINE (200,60)-(300,100),7,B
20 T$=CHR$(&H10)+CHR$(&H8)+CHR$(&H10)
30 T$=T$+CHR$(&H20)+CHR$(&H4)+CHR$(&H20)
40 T$=T$+CHR$(&H40)+CHR$(&H2)+CHR$(&H40)
50 T$=T$+CHR$(&H80)+CHR$(&H1)+CHR$(&H80)
60 PAINT (250,75),T$,7
```

次のように、波線でタイリングされます。



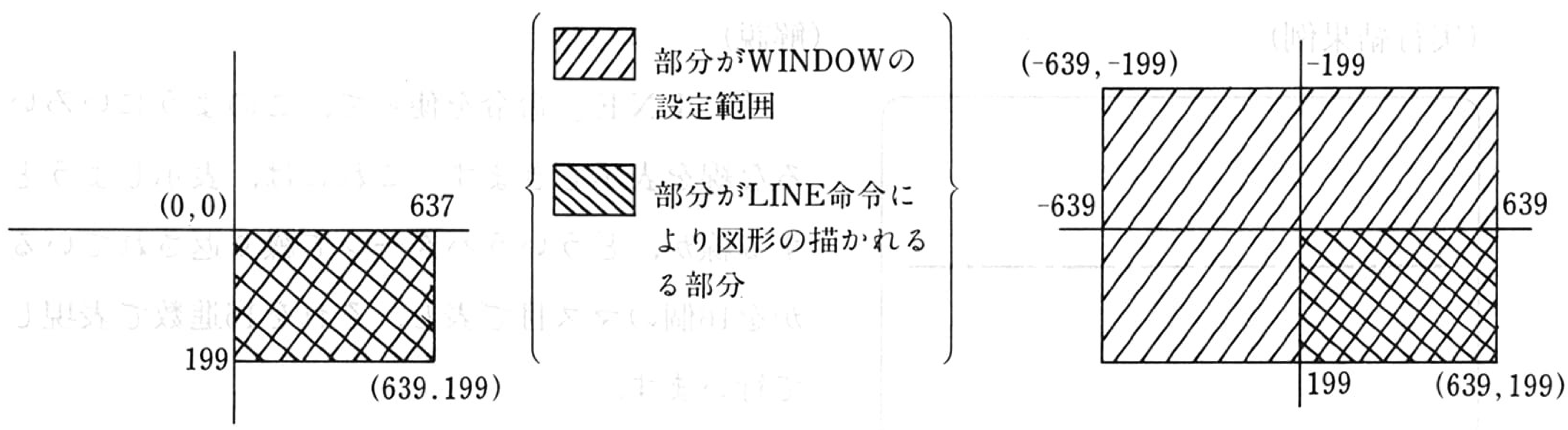
問題 4の解答例

```
10 WINDOW (-639,-199)-(639,199)
20 LINE (0,0)-(639,199),2,BF
```

（解説）

「ＷＩＮＤＯＷ」は，図形を描く領域を設定するものです。つまり，画面には「ＷＩＮＤＯＷ」の設定範囲の内側に描かれた図形が縮小または拡大されて表示されます。

（変更前のＷＩＮＤＯＷ）　　　　　　　　　（解答例で設定したＷＩＮＤＯＷ）



問題 5の解答例

● 〈問１〉の修正例

```
5 ON ERROR GOTO *ER
110 *ER
120 IF ERR=11 AND ERL=80 THEN BEEP:PRINT " ﾜﾘｻﾞﾝ ﾃﾞｷﾏｾﾝ":RESUME NEXT
130 ON ERROR GOTO 0
```

(プログラムリスト)

```
5 ON ERROR GOTO *ER
10 CONSOLE 0,25,1,1
20 INPUT "ｶｽﾞ A= ",A
30 IF A=0 THEN END
40 INPUT "ｶｽﾞ B= ",B
50 PRINT "(   ﾜ    ) ";A+B
60 PRINT "(   ｻ    ) ";A-B
70 PRINT "(  ｾｷ    ) ";A*B
80 PRINT "(  ｼｮｳ   ) ";A/B
90 PRINT "(ﾍﾞｷｼﾞｮｳ) ";A^B
100 PRINT:GOTO 20
110 *ER
120 IF ERR=11 AND ERL=80 THEN BEEP:PRINT " ﾜﾘｻﾞﾝ ﾃﾞｷﾏｾﾝ":RESUME NEXT
130 ON ERROR GOTO 0
```

(解説)

問題4で行ったエラー処理とここでの違いは，〔RESUME〕命令の使い方です。問題4でエラー処理を行ったのち，指定された行番号(添字の入力)から実行をやり直しますが，ここでは行番号の代わりに〔NEXT〕が指定されていますので，実行はエラーが起きた次の行（行番号90）から行われます。

● 〈問2〉の修正例

```
125 IF ERR=6  AND ERL=90 THEN BEEP:PRINT " ｵｰﾊﾞｰ ﾌﾛｰ   ":RESUME NEXT
```

(プログラムリスト)

```
5 ON ERROR GOTO *ER
10 CONSOLE 0,25,1,1
20 INPUT "ｶｽﾞ A= ",A
30 IF A=0 THEN END
40 INPUT "ｶｽﾞ B= ",B
50 PRINT "(   ﾜ    ) ";A+B
60 PRINT "(   ｻ    ) ";A-B
70 PRINT "(  ｾｷ    ) ";A*B
80 PRINT "(  ｼｮｳ   ) ";A/B
90 PRINT "(ﾍﾞｷｼﾞｮｳ) ";A^B
100 PRINT:GOTO 20
110 *ER
120 IF ERR=11 AND ERL=80 THEN BEEP:PRINT " ﾜﾘｻﾞﾝ ﾃﾞｷﾏｾﾝ":RESUME NEXT
125 IF ERR=6  AND ERL=90 THEN BEEP:PRINT " ｵｰﾊﾞｰ ﾌﾛｰ   ":RESUME NEXT
130 ON ERROR GOTO 0
```

Q & A

A & Q

エラーに関するQ＆A

Q—1　プログラムを実行したら「？Redo from start」のメッセージが出ましたが，何のことですか。

A—1　これは，数値型の変数を使ってデータを入力する命令（たとえば，INPUT A）を実行したときに，文字を入力するとこのようなメッセージが表示されます。このメッセージは，「もう一度データを入れなおしなさい」という意味です。つまり，正しいデータを単に入力しなおすだけでよいわけです。

Q—2　プログラムを実行したら，「Bad file number in 100」というエラーメッセージが表示されました。行番号100は，ファイルのオープンをしているところで間違ってはいないと思うのですが，どうしてなのでしょうか。

〈行番号100の内容〉

```
100　OPEN　"2：ABCD"　AS　＃4
```

A—2　これは，システムを起動する際の「How many files（0—15）？」に対して指定した値より大きなファイル番号を指定したためと考えられます。

もう一度，システムを立ち上げ直して「How many files（0—15）？」のところで，ファイル番号より大きな値を指定してやり直すか，またはファイル番号を小さな値に変更します。

Q—3　TAB関数を使って，画面への表示を行おうとしたら，「Subscript out of range」というエラーメッセージが表示されました。

キーインした命令は次のようなものですが，どうしてでしょうか。

```
?␣TAB␣（30）␣；␣A
```

A—3　これは，TAB関数を指定する際に「TAB」と「(30)」を離して指定したために「TAB」という配列とみなされてしまったからです。

配列は，通常「DIM文」で大きさを指定しますが，「DIM文」で指定しなくても11個（添字0～10）の大きさの配列が，プログラムで使用した際に自動的に確保されます。

したがって，御質問のような場合，「TAB」という配列が11個の大きさで確保されたために，「(30)」という指定が，「Subscript out of range」のエラーの原因となります。

ですから，TAB関数を使う場合は必ず「TAB」と位置指定（n）を離さずに使うようにして下さい。

Q—4　DATA文の定義行で，「Syntax error」が発生してしまいました。DATA文に間違いはないと思うのですがどうしてですか。

〈エラーの起きたDATA文〉

```
10  DATA  11,"XY"
20  DATA  25,"ABC"
```

A—4　DATA文には，文法的な誤りはないようです。それでは，そのデータを読み込むREAD文はどうなっていますか。もし，文字定数を読み込む変数が数値型のものだと，このエラーが発生します。

(例)　READ　A, B

B ← 数値型（整数，単精度，倍精度）の変数

Q—5　次のプログラムを実行したら，「Syntax error in 20」というエラーメッセージが表示されました。なぜですか。

〈プログラム〉

```
10  N=10
20  FOR  I=1  TO  N
30      PRINT  I
40  NEXT  I
```

A—5　プログラムをみても，「Syntax error」となるような文法的な誤りはありません。このようなときは，一つの可能性として行番号20と行番号30が一つの行になっていることが考えられます。それを確かめる方法としては，行番号20にカーソルを移動して RET キーを押し，次にカーソルが移動する行を確かめます。ここで，行番号40にカーソルが移動しているようだと行番号20と30は完全に一つになっています。このようなときには，次の方法で行を分割します。

〈行の分割〉

- 分割する場所へカーソルを移動し，SHIFT ⊕ INS DEL を押してインサート・モードにした後 CTRL ⊕ リ マ を押します。(すると，カーソル以降の文字が次の行に移動します)
- カーソル以降の文字は，メモリーに入っていないので RET キーを押してメモリーに入れます。

※　ここでは，行番号30の先頭にカーソルを移動してインサート・モード(SHIFT ⊕ INS DEL を押す)にしてから CTRL ⊕ リ マ を押した後，RET キーを押します。

行の分割が終ったら，再度プログラムを実行してみて下さい。今度はうまく行くはずです。

Q—6　プログラムを実行したら，「Syntax error in 10」というエラーメッセージが表示されました。しかし，行番号10はコメント文で文法エラーということはないと思うのですが。

A—6　コメント文を示す「'」(アポストロフィ)」記号は正しくついていますか。「'」がないと，BASICはその行を通常の命令として翻訳します。

Q—7　プログラムの実行中に，「Undefined line number in 950」というエラーメッセージが表示されました。そこで，行番号950を表示させたら正しく入っています。どうして，「Undefined〜」になるのですか。

A—7　「Undefined line number in 950」というエラーメッセージの解釈に誤解があるようです。このエラーメッセージは，「行番号950の中で使われている行番号が見つからない」という意味で，「行番号950が見つからない」という意味ではありません。

このエラーは，プログラム実行中に分岐命令の分岐先行番号(GOTO命令やGOSUB命令で指定する行番号) で指定した行がない場合に起こります。

なお，行番号の付け替えを行うためにRENUMコマンドを使ったときに「Undefined line in xxx」というメッセージが出る場合も同じ意味です。

このエラーが起こった場合は，分岐先として指定した行番号を正しい行番号に修正します。

Q—8　プログラムを実行中，「Illegal function call」エラーが起こりました。なぜですか。

A—8　「Illegal function call」というエラーは，付録のエラーメッセージ一覧をみると「ステートメントおよび関数において，機能の呼び方が誤っている」場合に起こることになっています。これだけでは，ちょっとわかりにくいので次に具体的にこのエラーとなるような使い方について示します。エラーの起きたプログラムと見比べてみて，該当するようでしたら正しく修正して下さい。

- 位置や桁数を使った関数で，負(−)の値が指定された場合
- 配列の添字に，負(−)の値が指定された場合
- ランダムのファイルの入出力でレコード番号に，負(−)の値が指定された場合
- 座標に，負(−)の値が指定された場合 (たとえば，「LINE」や「PUT」などです。)

Q—9　「Bad allocation table」というエラーが出ました。これは，どういう意味ですか。

A—9　これは，フロッピーディスクに記録されているファイルを管理している部分(FAT)が壊れてしまったときに発生するエラーです。このエラーは，復旧不能なエラーでたいていの場合，このエラーの出たフロッピーディスクは，フォーマットし直さない限り使用することはできません。なお，フロッピーディスク自身に傷がついてしまったものは，フォーマットしても使えません。

※　フロッピーディスク装置の電源を誤って途中で切ったりした後，電源を入れ直した場合もこのエラーが発生します。しかし，この場合は，リセットボタンを押してシステムを立ち上げ直すことにより復旧できます。

Q—10　「Rename across disks」というエラーはどういう意味ですか。

A—10　これは，NAME命令でファイル名の変更をしようとしたときに，旧ファイル名と新ファイル名のドライブ番号が違っている場合に起こります。(旧ファイル名と新ファイル名に，異なるドライブ番号を指定してはいけません)

単純な間違いですが，ドライブ１以外のファイル名の変更を行うときに，次のようにうっかりやってしまうことが多いようです。

NAME　"2：ABC" AS "XYZ"

Q—11　「RETURN without GOSUB」エラーが，プログラムの実行の終わりに必ず出てしまいます。どうしてでしょうか。

A—11　メインルーチンの終わりにEND命令が書かれていないのに，続けてサブルーチンが書かれているとメインルーチンの実行が終わった後，このサブルーチンを実行してしまいます。ところが，このサブルーチンは，GOSUB命令で実行していないので，RETURN命令の戻る場所が定まらないために，「RETURN without GOSUB」のエラーが発生します。

メインルーチン	
サブルーチン	"END" がないと，次のサブルーチンを実行して，「RETURN without GOSUB」になります。

また，サブルーチンの中でCLEAR命令を実行した場合，戻る場所を記憶しておくスタックまでクリアしてしまうため，このエラーが発生します。

画面およびプリンタに関するQ＆A

Q―12　「CLS」または「CMD CLS」命令以外の命令で画面をクリアすることができないのですか。

A―12　画面には，テキスト画面とグラフィック画面の二つがあります。この二つの画面を同時にクリアする場合は，「CLS 3」かまたは拡張命令の「CMD CLS 3」でしかできません。

しかし，テキスト画面，グラフィック画面をおのおの別々にクリアする場合には，次の命令で行うことができます。

〈テキスト画面〉

●ＰＲＩＮＴ　ＣＨＲ＄(１２)

〈グラフィック画面〉

●ＬＩＮＥ（０，０）－（６３９，１９９），０，ＢＦ

※　LINE命令を用いるとグラフィック画面の一部分をクリアするときにも使えます。

Q―13　プリンタで12行分改行するのにLPRINT命令を12回使っているのですが，ほかに簡単に12行分改行する方法はないでしょうか。

A―13　プリンタ（PC-8821／PC-8822／PC-8023）には，15行までなら一度に改行する命令を次のように指定できます。

ＬＰＲＩＮＴ　ＣＨＲ＄(&Ｈ１Ｆ)；ＣＨＲ＄(&Ｈ１０＋ｎ)；

この命令の「ｎ」の部分を「12」と指定すれば，12行分の改行指定が一度にできます。なお，「ｎ」で指定できる範囲は０～15です。

Q―14　汎用紙１枚に75～80行ぐらい印字したいのですが，どうしても１枚に66行までしか印字できません。どうしたらいいですか。

A―14　現在，改行の間隔は６行／インチになっています。このままだと１ページには66行しか印字できませんが，改行の間隔を縮めればそれだけ多くの行が１ページに納まります。

改行の間隔を変えるには，次の命令を実行します。

ＬＰＲＩＮＴ　ＣＨＲ＄(&Ｈ１Ｂ)；"Ｂ"；

すると，改行の間隔が８行／インチになり，１ページには88行まで印字できるようになります。しかし８行／インチの改行幅だと上下の文字の間隔が縮まり，印字結果が見づらくなります。できれば，６行／インチの改行幅で印字する方がよいでしょう。

なお，６行／インチの改行幅に戻すには，次の命令を実行します。

LPRINT CHR$(&H1B);"A";

改行幅を変更したら，必ずこの命令を実行して元に戻しておくようにします。

Q—15　汎用紙一枚に25行分印字し終えたら，次のページの先頭から次の25行を印字するようにしたいのですが，簡単に改ページを行うことはできませんか。

A—15　改ページを行う命令は，次のとおりです。

LPRINT CHR$(&HC);

すると，何行印字していようが自動的に改ページして，次のページの先頭から印字できるようになります。

Q—16　プリンタに印字するとき左から10桁を無条件に空けたいのですが，改行するたびに「LPRINT TAB (10)；」を実行させるのも面倒です。何かよい方法はありませんか。

A—16　プリンタで印字を行うときの一番左端の印字位置を「レフトマージン」といいます。PC-8821／PC-8822／PC-8023のプリンタでは，次の命令を使ってレフトマージンの位置を自由に変えることができます。

LPRINT CHR$(&H1B);"L";"nnn";

この命令の「nnn」を「010」と指定すれば，レフトマージンを解除する命令が実行されるまで，レフトマージンが10桁目にセットされます。

レフトマージンを解除するには，次の命令を実行します。

LPRINT CHR$(H1B);"L";"000";

Q—17　プリンタの改行方向を変えるには，どうすればよいのですか。

A—17　PC-8821／PC-8822／PC-8023のプリンタでは，次の命令を使うことによって，改行方向を自由に変えることができます。

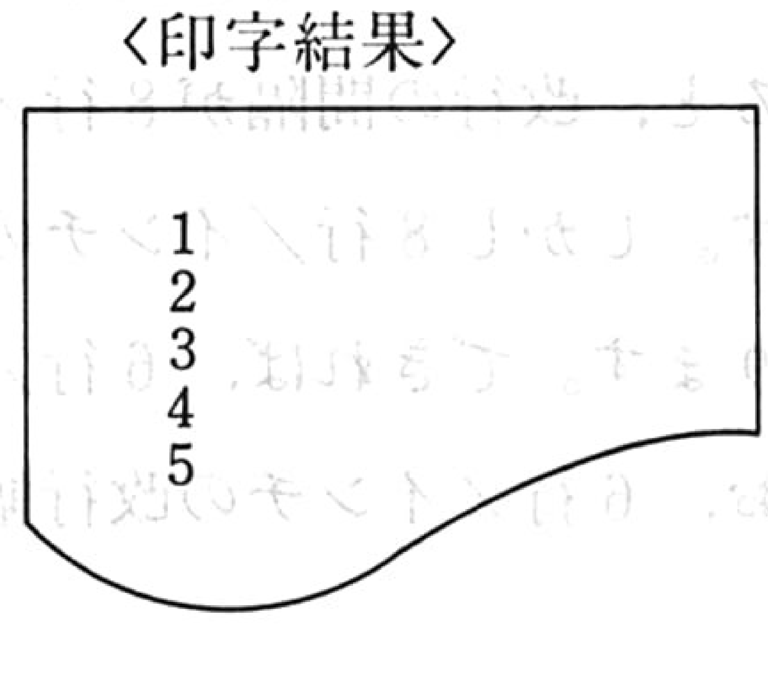
LPRINT CHR$(H1B);"r";
LPRINT CHR$(H1B);"f";

たとえば，次のプログラムを通常どおりに実行すると，右下の結果のようになります。

〈プログラム〉

```
10 FOR I=1 TO 5
20 LPRINT I
30 NEXT I
```

〈印字結果〉

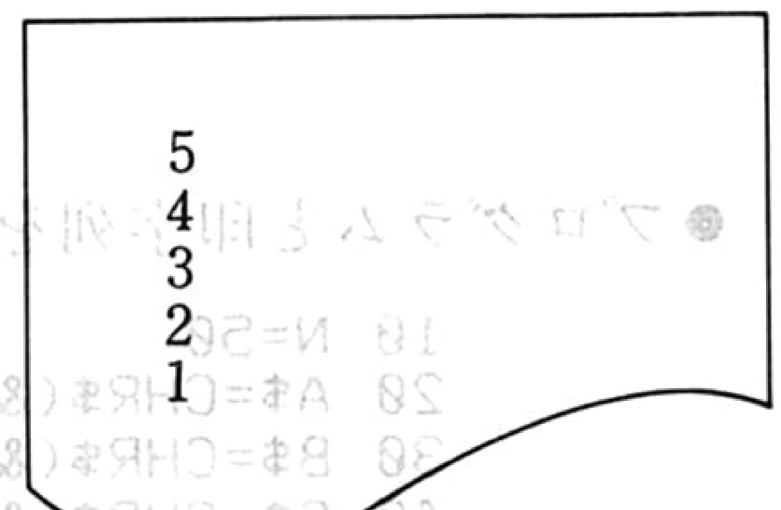


このプログラムの先頭で，次のように改行方向を逆にする命令を実行すると，結果は右下のようになります。

〈プログラム〉

```
5  LPRINT CHR$(H1B); "r";
10 FOR I=1 TO 5
20 LPRINT I
30 NEXT I
```

〈印字結果〉



元に戻すには，次の命令を実行します。

```
LPRINT CHR$(H1B); "f";
```

Q—18　プリンタに棒グラフを印字するときに，模様を付けることができますか。

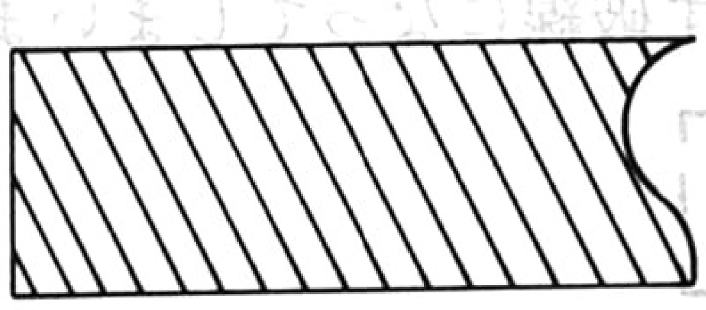
A—18　ドット列印字モードで，ドットイメージをプリントすることにより模様を印字することができます。

出力できるドットイメージは，PC-8821／PC-8822では16ビットと8ビットの二とおり，PC-8023では8ビットのイメージがあります。

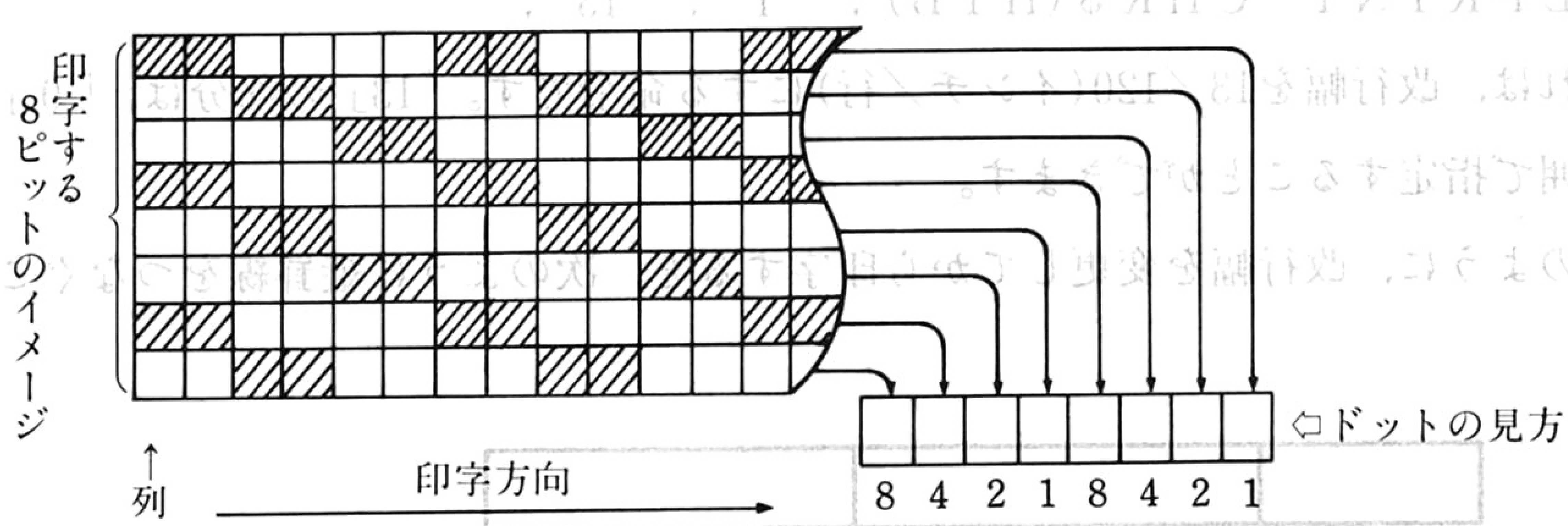
次に，斜線の模様を8ビットドットイメージで印字するときの例を示します。

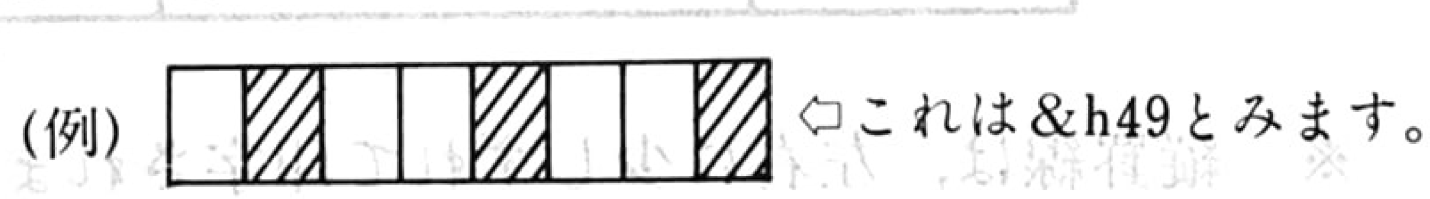
(例)

●印字する模様



●印字する模様のドットイメージ





●ドットイメージの印字は，列ごとの8ビットのイメージをプリンタに送って行います。

ここでの列ごとの8ビットイメージは，次のようになります。

列	1	2	3	4	5	6	7	8	9	…………
8ビットイメージ (16進数)	49	49	92	92	24	24	49	49	92	…………

●プリンタを8ビットドット印字モードにするためには，次の命令を実行します。

LPRINT　CHR$(H1B)；"S"；"0006"

（"0006"：送るドット列数）
（"S"：8ビットドット印字モード指定）

●プログラムと印字列を次に示します。

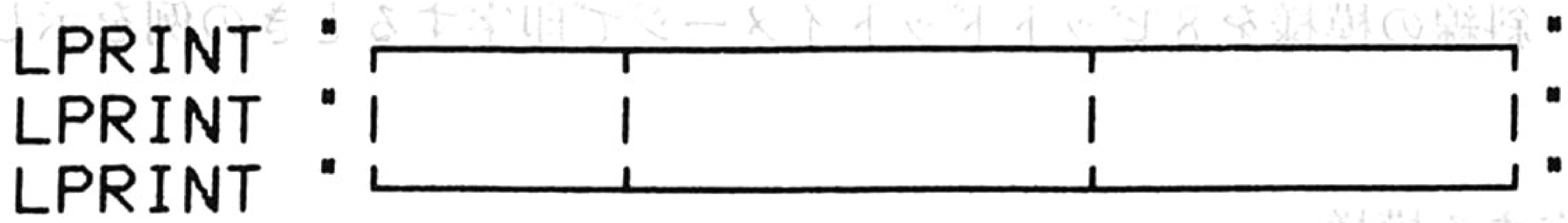
```
10 N=50
20 A$=CHR$(&H49)
30 B$=CHR$(&H92)
40 C$=CHR$(&H24)
50 FOR J=1 TO N
60   LPRINT CHR$(&H1B);"S";"0006";
70   LPRINT A$;A$;B$;B$;C$;C$;
80 NEXT J
```

（行60の"0006"：次のLPRINT命令で印字するドット列の数）
（行70の A$;A$;B$;B$;C$;C$;：これは，次のように書いても同じことになります。）

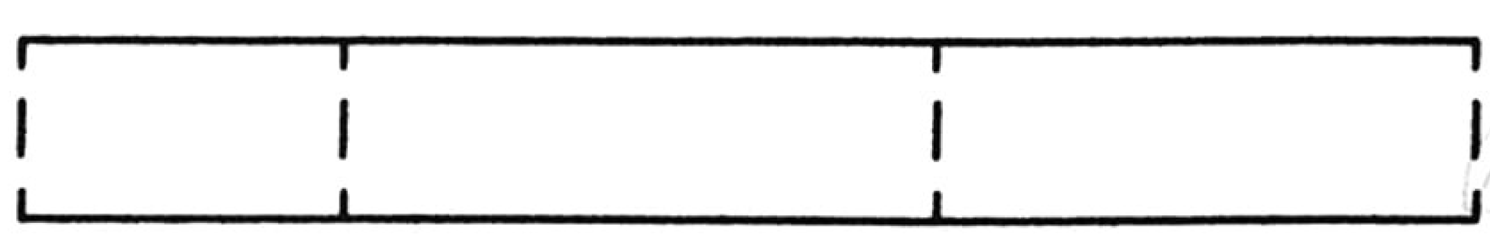
```
STRING$(2,A$);STRING$(2,B$);STRING$(2,C$);
```

Q—19　プリンタで罫線を印字するにはどうするのですか。

A—19　罫線を印字するには，グラフィック文字を使って，次のようにLPRINT命令で印字します。



しかし，このままでは次のように縦の罫線がつながらず破線になってしまいます。

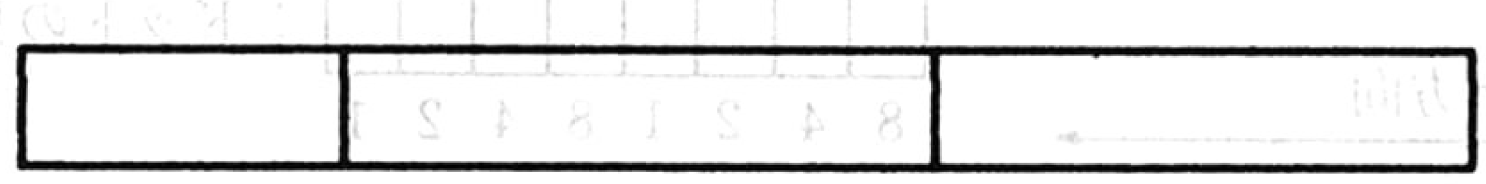


そこで，縦罫線をつなぐために次の命令で改行幅を変更します。

LPRINT　CHR$(H1B)；"T"；"13"；

これは，改行幅を13／120(インチ／行)にする命令です。「13」の部分は，「0」～「99」の範囲で指定することができます。

このように，改行幅を変更してから印字すると，次のように縦罫線をつなぐことができます。



※　縦罫線は，左右に少しずれて印字されますので，次の命令で，印字する方向を両方向印字から片方向印字に変更して，縦罫線の位置を揃えるようにします。

LPRINT　CHR$(H1B)；"［"；(PC-8023/Cの場合)

LPRINT　CHR$(H1B)；">"；(PC-8821/PC-8822の場合)

片方向印字から両方向印字に戻すには，次の命令を実行します。

LPRINT　CHR＄(H1B)；"]"；(共通)

以上のような機能を使うことによって，罫線を印字することができます。

なお，6行／インチの改行幅に戻すには，次の命令を実行します。

LPRINT　CHR＄(H1B)；"A"；

次に，罫線を印字する簡単なプログラムとその印字例を示します。

〈プログラム例〉

```
100 DIM X%(4),Y$(4),C(4)
110 A$="┌──────┬────────────┬──────────┐"
120 B$="│ コード │ショウヒンメイ      │  タ ン カ  │"
130 C$="╞══════╪════════════╪══════════╡"
140 D$="│      │            │          │"
150 E$="│######│&          &│¥¥###,###│"
160 F$="└──────┴────────────┴──────────┘"
170 FOR I=0 TO 4
180   INPUT "DATA=(%,$,!)";X%(I),Y$(I),C(I)
190 NEXT I
200 LPRINT CHR$(&H1B);"T";"13";
210 LPRINT CHR$(&H1B);"[";
220 LPRINT A$:LPRINT D$:LPRINT B$:LPRINT D$:LPRINT C$
230 FOR I=0 TO 4
240   LPRINT D$:LPRINT USING E$;X%(I);Y$(I);C(I)
250 NEXT I
260 LPRINT D$:LPRINT F$
270 LPRINT CHR$(&H1B);"]";
280 LPRINT CHR$(&H1B);"A";
```

〈印字例〉

コード	ショウヒンメイ	タ ン カ
10125	カラーテレビ	¥143,000
10631	ビデオ	¥215,000
20222	レイゾウコ	¥187,000
16111	センタクキ	¥43,000
25849	カンソウキ	¥21,000

フロッピーディスクに関するQ＆A

Q—20　フロッピーディスクを使っている間，システムディスクはずっとフロッピーディスク装置に入れたままにしておかなければなりませんか。

A—20　その必要はありません。PC-8801mkIIでフロッピーディスクを使う場合，システムが起動したらシステムディスクはすぐにでも取り出せます。ただし，すでにOPEN命令を実行してファイルをオープンしていたら，必ずCLOSE命令を実行してファイルをクローズした後に取り出すようにします。

なお，N-BASICの状態では，フロッピーディスクを取り出す前に必ずREMOVEコマンドを実行しておきます。

Q—21　ランダムファイルに書かれている最後のレコードの番号を調べるには，どうしたらよいですか。

A—21　LOF関数を使うことによって，簡単に調べることができます。

LOF関数は，ファイルをオープンしている間はいつでも使うことができます。また，新たに最後のレコードの後に追加を行った場合，LOF関数の値は変わります。

Q—22　形式の違ったデータファイルの内容を見たいのですが，それぞれのデータファイルの形式に合ったプログラムを作らなければならないのですか。

A—22　LINE INPUT#という命令を使えば，どんな形式のデータファイルでも読むことができます。

LINE INPUT#命令は，１件分のデータを一つの文字型変数に読み込みます。(ただし，１回の命令で読み込める文字の数は最大で255文字までです)

この命令を使うと，シーケンシャルファイルのデータの項目の区切り文字（「,」やスペース）まで読み込むことができます。

また，アスキーセーブされたプログラムファイルをシーケンシャルファイルとしてOPENし，このLINE INPUT#命令でそれぞれのデータ（プログラムの各行）を読み込むこともできます。

Q—23　フロッピーディスクにあとどのくらいデータが記録できるかを調べたいのですが，どうすればいいですか。

A—23　フロッピーディスクの未使用量を調べるには，DSKF関数を使って次のようにします。

PRINT　DSKF（ドライブ番号）

すると，ドライブ番号で指定されたフロッピーディスクの残りの容量が，次の例のようにクラスタ単位で表示されます。

(例)
```
PRINT DSKF(1)
 97
Ok
```

ドライブ1に入っているフロッピーディスクの残りの容量が「97」クラスタであることを表しています。

そのほか，DSKF関数には片面フロッピーディスクか両面フロッピーディスクかを調べる機能や，ディレクトリの位置を表す機能もあります。

Q—24　シーケンシャルファイルへ項目ごとに分けて入れたデータが，読み込むときは一つにつながってしまいます。どうしてなのでしょうか。

A—24　シーケンシャルファイルへデータを書き込むときのPRINT#命令で，項目の区切りとして「,」や「;」を使うと，各項目がスペースで区切られて書き込まれます。これを文字列変数に読み込むと，区切りとしてのスペースも文字データとして扱われるために，すべての項目が一つの文字データとしてつながってしまいます。

このような状態を避けるには，PRINT#命令の区切り記号（「,」や「;」を次の例のように，「; ","; 」に置き換えて項目の区切り文字として「,」を書き込むようにします。

(例)

PRINT　"ABC"; ","; "DEF"　⇦「ABC, DEF」と書き込まれます。

その他のQ＆A

Q―25　プログラムの実行速度を速くするような命令はありませんか。

A―25　プログラムの実行速度を速くする特別の命令は，N_{88}-BASICにはありません。しかし，テキスト画面への表示を行わない場合には，テキスト画面の表示を止める命令を実行することにより，プログラムの実行速度を速くすることができます。この命令は，拡張命令で次のようなものです。

●テキスト画面の表示を止める（その分だけ実行速度を速くできる）

```
CMD TEXT OFF
```

●テキスト画面の表示を再開する

```
CMD TEXT ON
```

この機能は，「CMD SING」命令の「X 0」というサブコマンドでも指定することができます。

また，画面に関して実行速度を速くするという点では，「SCREEN」命令の高速書き込みスイッチをONにすることで，グラフィック画面への図形の書き込みが速くなります。

高速書き込みスイッチをONにするには，「SCREEN ，1」か「SCREEN ，3」と指定します。高速書き込みスイッチをONにすると，図形を書き込む際に画面がチラつきます。そこで，図形を書く過程を見なくてもよい場合は，グラフィックマスクスイッチもONにして（SCREEN ，3）実行するようにします。

Q―26　拡張命令を使う場合は，プログラムを記憶する分のメモリーが減っていると聞きました。一度「@load.n88」で拡張命令をメモリーに展開するとその分のエリアは，システムを立ち上げ直さない限り使えないのでしょうか。

A―26　拡張命令が入っていたエリアを，ユーザーのプログラムで使えるようにするために「CMD CUT」という命令が拡張命令に用意されています。この命令を実行すると、拡張命令が使っていたエリアが開放されますが，これと同時に拡張命令の命令（CMD ×××）は使えなくなります。再度拡張命令を使う場合は「@load.n88」を実行して下さい。

Q—27 PUT@命令では，相対座標の指定（STEP）を行うことができません。PUT@命令で相対的な座標を指定する方法を教えて下さい。また，このほかに相対座標の指定ができないグラフィック命令がありますか。

A—27 N_{88}-BASICで相対座標の指定が許されているのは，次の命令です。

・PSET
・PRESET
・LINE
・CIRCLE
・PAINT
・POINT
・GET@

したがって，PUT@命令では「STEP」を付けることができません。しかし，次のように座標の指定を行うことで，現在の基点からの相対的な位置を指定することができます。

X座標＝POINT(2)＋X方向に対する相対位置
Y座標＝POINT(3)＋Y方向に対する相対位置

（例） 漢字をX，Y方向に20ドットずつずらして表示する場合

```
10  POINT (0,0)
20  FOR  I=1  TO  9
30      PUT@(POINT(2)+20, POINT(3)+20), KANJI(&H3246)
40  NEXT  I
```

また，PUT@命令以外で相対座標の指定ができないグラフィック命令としては，WINDOW，VIEWがあります。さらに，GET@の初めの（対角線の一方の）座標指定も相対座標の指定ができません。

Q—28　FOR〜NEXT命令で，初期値が最終値よりも大きいとき，その実行はどうなりますか。

A—28　N_{88}-BASICのFOR〜NEXT命令は，はじめから初期値が最終値より大きい場合は，実行は行われません。

なお，次のような場合も実行は行われません。

●増分値が正の値で初期値が最終値よりも大きい場合

（例）
```
FOR I=10 TO 1 STEP 1
 ∫
NEXT I
```

●増分値が負の値で初期値が最終値よりも小さい場合

（例）
```
FOR I=10 TO 20 STEP -1
 ∫
NEXT I
```

●初期値と最終値が等しく，かつ増分が「0」の場合

（例）
```
FOR I=10 TO 1 STEP 0
 ∫
NEXT I
```

Q—29　INPUT命令を実行したときに，RET キーだけを押すとどうなりますか。

A—29　値の入力を省略して RET キーだけを入力すると，その入力項目が数値変数の場合は「0」が，文字列変数の場合はヌルストリングが入ります。

Q—30　カーソルキー（↑，↓，←，→）を変数に入力する方法はありませんか。

A—30　カーソルキー等の特殊なキーを入力するには，INKEY$関数か，またはINPUT$関数を使います。

次に，INKEY$関数を使ってカーソルキーを入力しその方向に点を表示するプログラムの例を示します。

〈プログラム例〉

```
10 CONSOLE ,,,1:WIDTH 80,25
20 X=80:Y=50
30 LINE (X,Y)-(X+1,Y+1),7,BF
40 A$=INKEY$
50 IF A$="" THEN 40
60 IF A$=CHR$(28) THEN X=X+2:GOTO 100
70 IF A$=CHR$(29) THEN X=X-2:GOTO 100
80 IF A$=CHR$(30) THEN Y=Y-2:GOTO 100
90 IF A$=CHR$(31) THEN Y=Y+2:GOTO 100
100 LINE (X,Y)-(X+1,Y+1),7,BF
110 GOTO 40
```

※　このプログラムは，表示する点が画面の一番上か一番左にきたとき，さらにその方向にカーソルキーを押すと点が動かなくなります。

また，終了は STOP キーで行って下さい。

Q—31　BASICのプログラムの中から特定の文字列を探し出したいのですが，何かいい方法があったら教えて下さい。

A—31　まず，そのBASICプログラムを次のようにアスキー形式でフロッピーディスクにセーブしておきます。

SAVE　"ドライブ番号：プログラム名"，A

そこで，次のプログラムを実行し，セーブしたプログラムの名前と探し出そうとする文字列を指定します。

〈プログラム例〉

```
10 CLEAR 5000
20 N=300:DIM PL$(N)
30 INPUT "ﾌﾟﾛｸﾞﾗﾑ ﾒｲ ";PN$
40 OPEN PN$ FOR INPUT AS#1
50 FOR I=0 TO N
60   IF EOF(1) THEN 90
70   LINE INPUT#1,PL$(I)
80 NEXT I
90 CLOSE
100 LINE INPUT "ｻｶﾞｼﾀｲ ﾓｼﾞﾚﾂ ";S$
110 FOR J=0 TO I-1
120   IF INSTR(PL$(J),S$)><0 THEN PRINT PL$(J)
130 NEXT J
140 END
```

すると，プログラムの中から指定した特定の文字列を含む行をすべて画面に表示します。

Q—32　SQR関数を使って平方根を求めることはできますが，3乗根や4乗根はどうしたら求められますか。

A—32　指数関数のEXP関数と対数関数のLOG関数を使って，次のように求めます。

```
B=EXP (LOG(A)/N)
```

たとえば，「100」の3乗根を求める場合は，Aに「100」を与え，Nに「3」を与えて求めます。

なお，関数の計算は単精度で行われているために，結果に誤差が出る場合があります。

Q—33　常用対数を使いたいのですが，PC-8801 mkIIには自然対数を求める関数しかありません。常用対数を求める方法はないのですか。

A—33　常用対数を自然対数に置き換えると，次のようになります。

$$\log_{10}X = \frac{\log_e X}{\log_e 10}$$

したがって，このように自然対数に置き換えて計算すれば，自然対数の関数を使って常用対数を求めることができます。

たとえば，$Y=\log_{10}X$を求める場合

```
Y=LOG(X)/LOG(10)
```

のように書きます。

Q—34　SIN関数を使って次の命令を実行したところ，正しい真数が表示されませんがどうしてですか。

```
PRINT SIN (30)
```

A—34　三角関数は，ラジアンで指定しなければなりません。角度をラジアンに変換する計算式を次に示します。

角度＊π／180

したがって，SIN30°を求めるには次のように行います。

```
PRINT SIN (30*3.14159/180)
```

Q—35　パソコンで計算した合計と電卓で計算した合計の値が違います。どうしてですか。

A—35　現在の電卓は，ほとんどが8桁で計算します。ところが，パソコンの場合単精度で計算を行うと値が6桁に丸められるので，電卓よりも精度が低くなります。しかしこの計算を倍精度で行えば，電卓よりもはるかに精度が高くなります。

また，厳密に言うと，電卓の場合でも小数点の計算では誤差が生じます。小数点における誤差は，10進数で表現している小数をパソコン内部で2進数で表現しようとすると無限小数になる場合があるからです。

Q—36　PC-8801mkIIを使うと，テレビに電波障害のようなものが起こります。これは，なぜでしょうか。

A—36　これは，コンピュータ（PC-8801mkII）の発生する高周波がアンテナと電源を通してテレビに入ってしまうからです。このうち，電源から入る高周波については，高周波を除去するフィルターを組み込んだコンセントを使うことによって取り除くことができます。しかし，アンテナを通して入る高周波を取り除く具体的な対策はありません。(コンピュータから発生した高周波をシールドするためパソコンのテレビ側にアース線をつないだ金網を置くことによってかなり減少させることができますが，あまり現実的とはいえません)

このようなときは，コンピュータ（PC-8801mkII）とテレビをできるだけ離して使うことをおすすめしますが，具体的にどのぐらい離せば良いのかは使用状況によって変わります。

付　　録

付.1　キャラクタコード表

このキャラクタコード表は16進数での表現になっていて，「上位4ビット」を先に「下位4ビット」を後に書きます。たとえば，「K」を示すには，「&H4B」と表します。

上位4ビット→　／　下位4ビット↓

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0		D_E		0	@	P		p				ー	タ	ミ		
1	S_H	D_1	!	1	A	Q	a	q			。	ア	チ	ム		円
2	S_X	D_2	"	2	B	R	b	r			「	イ	ツ	メ		年
3	E_X	D_3	#	3	C	S	c	s			」	ウ	テ	モ		月
4	E_T	D_4	$	4	D	T	d	t			、	エ	ト	ヤ		日
5	E_Q	N_K	%	5	E	U	e	u			・	オ	ナ	ユ		時
6	A_K	S_N	&	6	F	V	f	v			ヲ	カ	ニ	ヨ		分
7	B_L	E_B	'	7	G	W	g	w			ァ	キ	ヌ	ラ		秒
8	B_S	C_N	(	8	H	X	h	x			ィ	ク	ネ	リ	♠	
9	H_T	E_M	)	9	I	Y	i	y			ゥ	ケ	ノ	ル	♥	
A	L_F	S_B	*	:	J	Z	j	z			ェ	コ	ハ	レ	♦	
B	H_M	E_C	+	;	K	[	k	{			ォ	サ	ヒ	ロ	♣	
C	C_L	→	,	<	L	¥	l	¦			ャ	シ	フ	ワ	●	
D	C_R	←	-	=	M	]	m	}			ュ	ス	ヘ	ン	○	
E	S_O	↑	.	>	N		n	~			ョ	セ	ホ	゛		
F	S_I	↓	/	?	O	_	o				ッ	ソ	マ	゜		

付.2 漢字コード一覧表

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
記号	212			、	。	，	．	・	：	；	？	！	゛	゜	´	｀	¨
	213	＾	￣	＿	ヽ	ヾ	ゝ	ゞ	〃	仝	々	〆	〇	ー	―	‐	／
	214	＼	～	∥	｜	…	‥	‘	’	“	”	（	）	〔	〕	［	］
	215	¦	¦	〈	〉	《	》	「	」	『	』	【	】	＋	－	±	×
	216	÷	＝	≠	＜	＞	≦	≧	∞	∴	♂	♀	°	′	″	℃	￥
	217	＄	￠	￡	％	＃	＆	＊	＠	§	☆	★	○	●	◎	◇	
	222		◆	□	■	△	▲	▽	▼	※	〒	→	←	↑	↓	〓	
英・数字	233	0	1	2	3	4	5	6	7	8	9						
	234		A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
	235	P	Q	R	S	T	U	V	W	X	Y	Z					
	236		a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
	237	p	q	r	s	t	u	v	w	x	y	z					
ひらがな	242		ぁ	あ	ぃ	い	ぅ	う	ぇ	え	ぉ	お	か	が	き	ぎ	く
	243	ぐ	け	げ	こ	ご	さ	ざ	し	じ	す	ず	せ	ぜ	そ	ぞ	た
	244	だ	ち	ぢ	っ	つ	づ	て	で	と	ど	な	に	ぬ	ね	の	は
	245	ば	ぱ	ひ	び	ぴ	ふ	ぶ	ぷ	へ	べ	ぺ	ほ	ぼ	ぽ	ま	み
	246	む	め	も	ゃ	や	ゅ	ゆ	ょ	よ	ら	り	る	れ	ろ	ゎ	わ
	247	ゐ	ゑ	を	ん												
カタカナ	252		ァ	ア	ィ	イ	ゥ	ウ	ェ	エ	ォ	オ	カ	ガ	キ	ギ	ク
	253	グ	ケ	ゲ	コ	ゴ	サ	ザ	シ	ジ	ス	ズ	セ	ゼ	ソ	ゾ	タ
	254	ダ	チ	ヂ	ッ	ツ	ヅ	テ	デ	ト	ド	ナ	ニ	ヌ	ネ	ノ	ハ
	255	バ	パ	ヒ	ビ	ピ	フ	ブ	プ	ヘ	ベ	ペ	ホ	ボ	ポ	マ	ミ
	256	ム	メ	モ	ャ	ヤ	ュ	ユ	ョ	ヨ	ラ	リ	ル	レ	ロ	ヮ	ワ
	257	ヰ	ヱ	ヲ	ン	ヴ	ヵ	ヶ									
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
半角文字	002		!	"	#	$	%	&	'	(	)	*	+	,	−	.	/
	003	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
	004	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
	005	P	Q	R	S	T	U	V	W	X	Y	Z	[	¥	]	^	—
	006		a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
	007	p	q	r	s	t	u	v	w	x	y	z	{	¦	}	~	
	008		。	「	」	、	・	を	ぁ	ぃ	ぅ	ぇ	ぉ	ゃ	ゅ	ょ	っ
	009	ー	あ	い	う	え	お	か	き	く	け	こ	さ	し	す	せ	そ
	00A		。	「	」		・	ヲ	ァ	ィ	ゥ	ェ	ォ	ャ	ュ	ョ	ッ
	00B	ー	ア	イ	ウ	エ	オ	カ	キ	ク	ケ	コ	サ	シ	ス	セ	ソ
	00C	タ	チ	ツ	テ	ト	ナ	ニ	ヌ	ネ	ノ	ハ	ヒ	フ	ヘ	ホ	マ
	00D	ミ	ム	メ	モ	ヤ	ユ	ヨ	ラ	リ	ル	レ	ロ	ワ	ン	゛	゜
	00E	た	ち	つ	て	と	な	に	ぬ	ね	の	は	ひ	ふ	へ	ほ	ま
	00F	み	む	め	も	や	ゆ	よ	ら	り	る	れ	ろ	わ	ん	゛	゜
¼角文字	010		SH	SX	EX	ET	EQ	AK	BL	BS	HT	LF	HM	CL	CR	SO	SI
	011	DE	D1	D2	D3	D4	NK	SN	EB	CN	EM	SB	EC	→	←	↑	↓
	012		!	"	#	$	%	&	'	(	)	*	+	,	−	.	/
	013	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
	014	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
	015	P	Q	R	S	T	U	V	W	X	Y	Z	[	¥	]	∧	—
	016		a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
	017	p	q	r	s	t	u	v	w	x	y	z	{	¦	}	~	
	018	▁	▂	▃	▄	▅	▆	▇	█	▏	▎	▍	▌	▋	▊	▉	┼
	019	┴	┬	┤	├	▔	▔	▕	▕	┌	┐	└	┘	╭	╮	╰	╯
	01A		。	「	」	、	・	ヲ	ァ	ィ	ゥ	ェ	ォ	ャ	ュ	ョ	ッ
	01B	ー	ア	イ	ウ	エ	オ	カ	キ	ク	ケ	コ	サ	シ	ス	セ	ソ
	01C	タ	チ	ツ	テ	ト	ナ	ニ	ヌ	ネ	ノ	ハ	ヒ	フ	ヘ	ホ	マ
	01D	ミ	ム	メ	モ	ヤ	ユ	ヨ	ラ	リ	ル	レ	ロ	ワ	ン	゛	゜
	01E	═	╞	╪	╡	◢	◣	◥	◤	♠	♥	♦	♣	●	○	╱	╲
	01F	×	円	年	月	日	時	分	秒								
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
ギリシア文字	262		Α	Β	Γ	Δ	Ε	Ζ	Η	Θ	Ι	Κ	Λ	Μ	Ν	Ξ	Ο
	263	Π	Ρ	Σ	Τ	Υ	Φ	Χ	Ψ	Ω							
	264		α	β	γ	δ	ε	ζ	η	θ	ι	κ	λ	μ	ν	ξ	ο
	265	π	ρ	σ	τ	υ	φ	χ	ψ	ω							
ロシア文字	272		А	Б	В	Г	Д	Е	Ё	Ж	З	И	Й	К	Л	М	Н
	273	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э
	274	Ю	Я														
	275		а	б	в	г	д	е	ё	ж	з	и	й	к	л	м	н
	276	о	п	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э
	277	ю	я														
ア	302		亜	唖	娃	阿	哀	愛	挨	姶	逢	葵	茜	穐	悪	握	渥
	303	旭	葦	芦	鯵	梓	圧	斡	扱	宛	姐	虻	飴	絢	綾	鮎	或
	304	粟	袷	安	庵	按	暗	案	闇	鞍	杏						
イ	304											以	伊	位	依	偉	囲
	305	夷	委	威	尉	惟	意	慰	易	椅	為	畏	異	移	維	緯	胃
	306	萎	衣	謂	違	遺	医	井	亥	域	育	郁	磯	一	壱	溢	逸
	307	稲	茨	芋	鰯	允	印	咽	員	因	姻	引	飲	淫	胤	蔭	
	312		院	陰	隠	韻	吋										
ウ	312							右	宇	烏	羽	迂	雨	卯	鵜	窺	丑
	313	碓	臼	渦	嘘	唄	欝	蔚	鰻	姥	厩	浦	瓜	閏	噂	云	運
	314	雲															
エ	314		荏	餌	叡	営	嬰	影	映	曳	栄	永	泳	洩	瑛	盈	穎
	315	頴	英	衛	詠	鋭	液	疫	益	駅	悦	謁	越	閲	榎	厭	円
	316	園	堰	奄	宴	延	怨	掩	援	沿	演	炎	焔	煙	燕	猿	縁
	317	艶	苑	薗	遠	鉛	鴛	塩									
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
オ	317								於	汚	甥	凹	央	奥	往	応	
	322		押	旺	横	欧	殴	王	翁	襖	鴬	鴎	黄	岡	沖	荻	億
	323	屋	憶	臆	桶	牡	乙	俺	卸	恩	温	穏	音				
カ	323													下	化	仮	何
	324	伽	価	佳	加	可	嘉	夏	嫁	家	寡	科	暇	果	架	歌	河
	325	火	珂	禍	禾	稼	箇	花	苛	茄	荷	華	菓	蝦	課	嘩	貨
	326	迦	過	霞	蚊	俄	峨	我	牙	画	臥	芽	蛾	賀	雅	餓	駕
	327	介	会	解	回	塊	壊	廻	快	怪	悔	恢	懐	戒	拐	改	
	332		魁	晦	械	海	灰	界	皆	絵	芥	蟹	開	階	貝	凱	劾
	333	外	咳	害	崖	慨	概	涯	碍	蓋	街	該	鎧	骸	浬	馨	蛙
	334	垣	柿	蛎	鈎	劃	嚇	各	廓	拡	撹	格	核	殻	獲	確	穫
	335	覚	角	赫	較	郭	閣	隔	革	学	岳	楽	額	顎	掛	笠	樫
	336	橿	梶	鰍	潟	割	喝	恰	括	活	渇	滑	葛	褐	轄	且	鰹
	337	叶	椛	樺	鞄	株	兜	竃	蒲	釜	鎌	噛	鴨	栢	茅	萱	
	342		粥	刈	苅	瓦	乾	侃	冠	寒	刊	勘	勧	巻	喚	堪	姦
	343	完	官	寛	干	幹	患	感	慣	憾	換	敢	柑	桓	棺	款	歓
	344	汗	漢	澗	潅	環	甘	監	看	竿	管	簡	緩	缶	翰	肝	艦
	345	莞	観	諌	貫	還	鑑	間	閑	関	陥	韓	館	舘	丸	含	岸
	346	巌	玩	癌	眼	岩	翫	贋	雁	頑	顔	願					
キ	346												企	伎	危	喜	器
	347	基	奇	嬉	寄	岐	希	幾	忌	揮	机	旗	既	期	棋	棄	
	352		機	帰	毅	気	汽	畿	祈	季	稀	紀	徽	規	記	貴	起
	353	軌	輝	飢	騎	鬼	亀	偽	儀	妓	宜	戯	技	擬	欺	犠	疑
	354	祇	義	蟻	誼	議	掬	菊	鞠	吉	吃	喫	桔	橘	詰	砧	杵
	355	黍	却	客	脚	虐	逆	丘	久	仇	休	及	吸	宮	弓	急	救
	356	朽	求	汲	泣	灸	球	究	窮	笈	級	糾	給	旧	牛	去	居
	357	巨	拒	拠	挙	渠	虚	許	距	鋸	漁	禦	魚	亨	享	京	
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	362		供	侠	僑	兇	競	共	凶	協	匡	卿	叫	喬	境	峡	強
	363	彊	怯	恐	恭	挟	教	橋	況	狂	狭	矯	胸	脅	興	蕎	郷
	364	鏡	響	饗	驚	仰	凝	尭	暁	業	局	曲	極	玉	桐	粁	僅
	365	勤	均	巾	錦	斤	欣	欽	琴	禁	禽	筋	緊	芹	菌	衿	襟
	366	謹	近	金	吟	銀											
ク	366						九	倶	句	区	狗	玖	矩	苦	躯	駆	駈
	367	駒	具	愚	虞	喰	空	偶	寓	遇	隅	串	櫛	釧	屑	屈	
	372		掘	窟	沓	靴	轡	窪	熊	隈	粂	栗	繰	桑	鍬	勲	君
	373	薫	訓	群	軍	郡											
ケ	373						卦	袈	祁	係	傾	刑	兄	啓	圭	珪	型
	374	契	形	径	恵	慶	慧	憩	掲	携	敬	景	桂	渓	畦	稽	系
	375	経	継	繋	罫	茎	荊	蛍	計	詣	警	軽	頚	鶏	芸	迎	鯨
	376	劇	戟	撃	激	隙	桁	傑	欠	決	潔	穴	結	血	訣	月	件
	377	倹	倦	健	兼	券	剣	喧	圏	堅	嫌	建	憲	懸	拳	捲	
	382		検	権	牽	犬	献	研	硯	絹	県	肩	見	謙	賢	軒	遣
	383	鍵	険	顕	験	鹸	元	原	厳	幻	弦	減	源	玄	現	絃	舷
	384	言	諺	限													
コ	384				乎	個	古	呼	固	姑	孤	己	庫	弧	戸	故	枯
	385	湖	狐	糊	袴	股	胡	菰	虎	誇	跨	鈷	雇	顧	鼓	五	互
	386	伍	午	呉	吾	娯	後	御	悟	梧	檎	瑚	碁	語	誤	護	醐
	387	乞	鯉	交	佼	侯	候	倖	光	公	功	効	勾	厚	口	向	
	392		后	喉	坑	垢	好	孔	孝	宏	工	巧	巷	幸	広	庚	康
	393	弘	恒	慌	抗	拘	控	攻	昂	晃	更	杭	校	梗	構	江	洪
	394	浩	港	溝	甲	皇	硬	稿	糠	紅	紘	絞	綱	耕	考	肯	肱
	395	腔	膏	航	荒	行	衡	講	貢	購	郊	酵	鉱	礦	鋼	閤	降
	396	項	香	高	鴻	剛	劫	号	合	壕	拷	濠	豪	轟	麹	克	刻
	397	告	国	穀	酷	鵠	黒	獄	漉	腰	甑	忽	惚	骨	狛	込	
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	3A2		此	頃	今	困	坤	墾	婚	恨	懇	昏	昆	根	梱	混	痕
	3A3	紺	艮	魂													
サ	3A3				些	佐	叉	唆	嵯	左	差	査	沙	瑳	砂	詐	鎖
	3A4	裟	坐	座	挫	債	催	再	最	哉	塞	妻	宰	彩	才	採	栽
	3A5	歳	済	災	采	犀	砕	砦	祭	斎	細	菜	裁	載	際	剤	在
	3A6	材	罪	財	冴	坂	阪	堺	榊	肴	咲	崎	埼	碕	鷺	作	削
	3A7	咋	搾	昨	朔	柵	窄	策	索	錯	桜	鮭	笹	匙	冊	刷	
	3B2		察	拶	撮	擦	札	殺	薩	雑	皐	鯖	捌	錆	鮫	皿	晒
	3B3	三	傘	参	山	惨	撒	散	桟	燦	珊	産	算	纂	蚕	讃	賛
	3B4	酸	餐	斬	暫	残											
シ	3B4						仕	仔	伺	使	刺	司	史	嗣	四	士	始
	3B5	姉	姿	子	屍	市	師	志	思	指	支	孜	斯	施	旨	枝	止
	3B6	死	氏	獅	祉	私	糸	紙	紫	肢	脂	至	視	詞	詩	試	誌
	3B7	諮	資	賜	雌	飼	歯	事	似	侍	児	字	寺	慈	持	時	
	3C2		次	滋	治	爾	璽	痔	磁	示	而	耳	自	蒔	辞	汐	鹿
	3C3	式	識	鴫	竺	軸	宍	雫	七	叱	執	失	嫉	室	悉	湿	漆
	3C4	疾	質	実	蔀	篠	偲	柴	芝	屡	蕊	縞	舎	写	射	捨	赦
	3C5	斜	煮	社	紗	者	謝	車	遮	蛇	邪	借	勺	尺	杓	灼	爵
	3C6	酌	釈	錫	若	寂	弱	惹	主	取	守	手	朱	殊	狩	珠	種
	3C7	腫	趣	酒	首	儒	受	呪	寿	授	樹	綬	需	囚	収	周	
	3D2		宗	就	州	修	愁	拾	洲	秀	秋	終	繍	習	臭	舟	蒐
	3D3	衆	襲	讐	蹴	輯	週	酋	酬	集	醜	什	住	充	十	従	戎
	3D4	柔	汁	渋	獣	縦	重	銃	叔	夙	宿	淑	祝	縮	粛	塾	熟
	3D5	出	術	述	俊	峻	春	瞬	竣	舜	駿	准	循	旬	楯	殉	淳
	3D6	準	潤	盾	純	巡	遵	醇	順	処	初	所	暑	曙	渚	庶	緒
	3D7	署	書	薯	藷	諸	助	叙	女	序	徐	恕	鋤	除	傷	償	
	3E2		勝	匠	升	召	哨	商	唱	嘗	奨	妾	娼	宵	将	小	少
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	3E3	尚	庄	床	廠	彰	承	抄	招	掌	捷	昇	昌	昭	晶	松	梢
	3E4	樟	樵	沼	消	渉	湘	焼	焦	照	症	省	硝	礁	祥	称	章
	3E5	笑	粧	紹	肖	菖	蒋	蕉	衝	裳	訟	証	詔	詳	象	賞	醤
	3E6	鉦	鍾	鐘	障	鞘	上	丈	丞	乗	冗	剰	城	場	壌	嬢	常
	3E7	情	擾	条	杖	浄	状	畳	穣	蒸	譲	醸	錠	嘱	埴	飾	
	3F2		拭	植	殖	燭	織	職	色	触	食	蝕	辱	尻	伸	信	侵
	3F3	唇	娠	寝	審	心	慎	振	新	晋	森	榛	浸	深	申	疹	真
	3F4	神	秦	紳	臣	芯	薪	親	診	身	辛	進	針	震	人	仁	刃
	3F5	塵	壬	尋	甚	尽	腎	訊	迅	陣	靭						
ス	3F5											笥	諏	須	酢	図	厨
	3F6	逗	吹	垂	帥	推	水	炊	睡	粋	翠	衰	遂	酔	錐	錘	随
	3F7	瑞	髄	崇	嵩	数	枢	趨	雛	据	杉	椙	菅	頗	雀	裾	
	402		澄	摺	寸												
セ	402					世	瀬	畝	是	凄	制	勢	姓	征	性	成	政
	403	整	星	晴	棲	栖	正	清	牲	生	盛	精	聖	声	製	西	誠
	404	誓	請	逝	醒	青	静	斉	税	脆	隻	席	惜	戚	斥	昔	析
	405	石	積	籍	績	脊	責	赤	跡	蹟	碩	切	拙	接	摂	折	設
	406	窃	節	説	雪	絶	舌	蝉	仙	先	千	占	宣	専	尖	川	戦
	407	扇	撰	栓	栴	泉	浅	洗	染	潜	煎	煽	旋	穿	箭	線	
	412		繊	羨	腺	舛	船	薦	詮	賎	践	選	遷	銭	銑	閃	鮮
	413	前	善	漸	然	全	禅	繕	膳	糎							
ソ	413										噌	塑	岨	措	曾	曽	楚
	414	狙	疏	疎	礎	祖	租	粗	素	組	蘇	訴	阻	遡	鼠	僧	創
	415	双	叢	倉	喪	壮	奏	爽	宋	層	匝	惣	想	捜	掃	挿	掻
	416	操	早	曹	巣	槍	槽	漕	燥	争	痩	相	窓	糟	総	綜	聡
	417	草	荘	葬	蒼	藻	装	走	送	遭	鎗	霜	騒	像	増	憎	
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	422		臓	蔵	贈	造	促	側	則	即	息	捉	束	測	足	速	俗
	423	属	賊	族	続	卒	袖	其	揃	存	孫	尊	損	村	遜		
タ	423															他	多
	424	太	汰	詑	唾	堕	妥	惰	打	柁	舵	楕	陀	駄	騨	体	堆
	425	対	耐	岱	帯	待	怠	態	戴	替	泰	滞	胎	腿	苔	袋	貸
	426	退	逮	隊	黛	鯛	代	台	大	第	醍	題	鷹	滝	瀧	卓	啄
	427	宅	托	択	拓	沢	濯	琢	託	鐸	濁	諾	茸	凧	蛸	只	
	432		叩	但	達	辰	奪	脱	巽	竪	辿	棚	谷	狸	鱈	樽	誰
	433	丹	単	嘆	坦	担	探	旦	歎	淡	湛	炭	短	端	箪	綻	耽
	434	胆	蛋	誕	鍛	団	壇	弾	断	暖	檀	段	男	談			
チ	434														値	知	地
	435	弛	恥	智	池	痴	稚	置	致	蜘	遅	馳	築	畜	竹	筑	蓄
	436	逐	秩	窒	茶	嫡	着	中	仲	宙	忠	抽	昼	柱	注	虫	衷
	437	註	酎	鋳	駐	樗	瀦	猪	苧	著	貯	丁	兆	凋	喋	寵	
	442		帖	帳	庁	弔	張	彫	徴	懲	挑	暢	朝	潮	牒	町	眺
	443	聴	脹	腸	蝶	調	諜	超	跳	銚	長	頂	鳥	勅	捗	直	朕
	444	沈	珍	賃	鎮	陳											
ツ	444						津	墜	椎	槌	追	鎚	痛	通	塚	栂	掴
	445	槻	佃	漬	柘	辻	蔦	綴	鍔	椿	潰	坪	壷	嬬	紬	爪	吊
	446	釣	鶴														
テ	446			亭	低	停	偵	剃	貞	呈	堤	定	帝	底	庭	廷	弟
	447	悌	抵	挺	提	梯	汀	碇	禎	程	締	艇	訂	諦	蹄	逓	
	452		邸	鄭	釘	鼎	泥	摘	擢	敵	滴	的	笛	適	鏑	溺	哲
	453	徹	撤	轍	迭	鉄	典	填	天	展	店	添	纒	甜	貼	転	顛
	454	点	伝	殿	澱	田	電										
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
ト	454							兎	吐	堵	塗	妬	屠	徒	斗	杜	渡
	455	登	菟	賭	途	都	鍍	砥	礪	努	度	土	奴	怒	倒	党	冬
	456	凍	刀	唐	塔	塘	套	宕	島	嶋	悼	投	搭	東	桃	檮	棟
	457	盗	淘	湯	濤	灯	燈	当	痘	禱	等	答	筒	糖	統	到	
	462		董	蕩	藤	討	謄	豆	踏	逃	透	鐙	陶	頭	騰	闘	働
	463	動	同	堂	導	憧	撞	洞	瞳	童	胴	萄	道	銅	峠	鴇	匿
	464	得	徳	瀆	特	督	禿	篤	毒	独	読	栃	橡	凸	突	椴	届
	465	鳶	苫	寅	酉	瀞	噸	屯	惇	敦	沌	豚	遁	頓	呑	曇	鈍
ナ	466	奈	那	内	乍	凪	薙	謎	灘	捺	鍋	楢	馴	縄	畷	南	楠
	467	軟	難	汝													
ニ	467				二	尼	弐	迩	匂	賑	肉	虹	廿	日	乳	入	
	472		如	尿	韮	任	妊	忍	認								
ヌ	472									濡							
ネ	472										禰	祢	寧	葱	猫	熱	年
	473	念	捻	撚	燃	粘											
ノ	473						乃	廼	之	埜	嚢	悩	濃	納	能	脳	膿
	474	農	覗	蚤													
ハ	474				巴	把	播	覇	杷	波	派	琶	破	婆	罵	芭	馬
	475	俳	廃	拝	排	敗	杯	盃	牌	背	肺	輩	配	倍	培	媒	梅
	476	楳	煤	狽	買	売	賠	陪	這	蝿	秤	矧	萩	伯	剥	博	拍
	477	柏	泊	白	箔	粕	舶	薄	迫	曝	漠	爆	縛	莫	駁	麦	
	482		函	箱	硲	箸	肇	筈	櫨	幡	肌	畑	畠	八	鉢	溌	発
	483	醗	髪	伐	罰	抜	筏	閥	鳩	噺	塙	蛤	隼	伴	判	半	反
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	484	叛	帆	搬	斑	板	氾	汎	版	犯	班	畔	繁	般	藩	販	範
	485	釆	煩	頒	飯	挽	晩	番	盤	磐	蕃	蛮					
ヒ	485												匪	卑	否	妃	庇
	486	彼	悲	扉	批	披	斐	比	泌	疲	皮	碑	秘	緋	罷	肥	被
	487	誹	費	避	非	飛	樋	簸	備	尾	微	枇	毘	琵	眉	美	
	492		鼻	柊	稗	匹	疋	髭	彦	膝	菱	肘	弼	必	畢	筆	逼
	493	桧	姫	媛	紐	百	謬	俵	彪	標	氷	漂	瓢	票	表	評	豹
	494	廟	描	病	秒	苗	錨	鋲	蒜	蛭	鰭	品	彬	斌	浜	瀕	貧
	495	賓	頻	敏	瓶												
フ	495					不	付	埠	夫	婦	富	冨	布	府	怖	扶	敷
	496	斧	普	浮	父	符	腐	膚	芙	譜	負	賦	赴	阜	附	侮	撫
	497	武	舞	葡	蕪	部	封	楓	風	葺	蕗	伏	副	復	幅	服	
	4A2		福	腹	複	覆	淵	弗	払	沸	仏	物	鮒	分	吻	噴	墳
	4A3	憤	扮	焚	奮	粉	糞	紛	雰	文	聞						
ヘ	4A3											丙	併	兵	塀	幣	平
	4A4	弊	柄	並	蔽	閉	陛	米	頁	僻	壁	癖	碧	別	瞥	蔑	箆
	4A5	偏	変	片	篇	編	辺	返	遍	便	勉	娩	弁	鞭			
ホ	4A5														保	舗	鋪
	4A6	圃	捕	歩	甫	補	輔	穂	募	墓	慕	戊	暮	母	簿	菩	倣
	4A7	俸	包	呆	報	奉	宝	峰	峯	崩	庖	抱	捧	放	方	朋	
	4B2		法	泡	烹	砲	縫	胞	芳	萌	蓬	蜂	褒	訪	豊	邦	鋒
	4B3	飽	鳳	鵬	乏	亡	傍	剖	坊	妨	帽	忘	忙	房	暴	望	某
	4B4	棒	冒	紡	肪	膨	謀	貌	貿	鉾	防	吠	頬	北	僕	卜	墨
	4B5	撲	朴	牧	睦	穆	釦	勃	没	殆	堀	幌	奔	本	翻	凡	盆
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
マ	4B6	摩	磨	魔	麻	埋	妹	昧	枚	毎	哩	槙	幕	膜	枕	鮪	柾
	4B7	鱒	桝	亦	俣	又	抹	末	沫	迄	儘	繭	麿	万	慢	満	
	4C2		漫	蔓													
ミ	4C2				味	未	魅	巳	箕	岬	密	蜜	湊	蓑	稔	脈	妙
	4C3	粍	民	眠													
ム	4C3				務	夢	無	牟	矛	霧	鵡	椋	婿	娘			
メ	4C3														冥	名	命
	4C4	明	盟	迷	銘	鳴	姪	牝	滅	免	棉	綿	緬	面	麺		
モ	4C4															摸	模
	4C5	茂	妄	孟	毛	猛	盲	網	耗	蒙	儲	木	黙	目	杢	勿	餅
	4C6	尤	戻	籾	貰	問	悶	紋	門	匁							
ヤ	4C6										也	冶	夜	爺	耶	野	弥
	4C7	矢	厄	役	約	薬	訳	躍	靖	柳	薮	鑓					
ユ	4C7												愉	愈	油	癒	
	4D2		諭	輸	唯	佑	優	勇	友	宥	幽	悠	憂	揖	有	柚	湧
	4D3	涌	猶	猷	由	祐	裕	誘	遊	邑	郵	雄	融	夕			
ヨ	4D3														予	余	与
	4D4	誉	輿	預	傭	幼	妖	容	庸	揚	揺	擁	曜	楊	様	洋	溶
	4D5	熔	用	窯	羊	耀	葉	蓉	要	謡	踊	遥	陽	養	慾	抑	欲
	4D6	沃	浴	翌	翼	淀											
ラ	4D6						羅	螺	裸	来	莱	頼	雷	洛	絡	落	酪
	4D7	乱	卵	嵐	欄	濫	藍	蘭	覧								
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
リ	4D7									利	吏	履	李	梨	理	璃	
	4E2		痢	裏	裡	里	離	陸	律	率	立	葎	掠	略	劉	流	溜
	4E3	琉	留	硫	粒	隆	竜	龍	侶	慮	旅	虜	了	亮	僚	両	凌
	4E4	寮	料	梁	涼	猟	療	瞭	稜	糧	良	諒	遼	量	陵	領	力
	4E5	緑	倫	厘	林	淋	燐	琳	臨	輪	隣	鱗	麟				
ル	4E5													瑠	塁	涙	累
	4E6	類															
レ	4E6		令	伶	例	冷	励	嶺	怜	玲	礼	苓	鈴	隷	零	霊	麗
	4E7	齢	暦	歴	列	劣	烈	裂	廉	恋	憐	漣	煉	簾	練	聯	
	4F2		蓮	連	錬												
ロ	4F2					呂	魯	櫓	炉	賂	路	露	労	婁	廊	弄	朗
	4F3	楼	榔	浪	漏	牢	狼	籠	老	聾	蝋	郎	六	麓	禄	肋	録
	4F4	論															
ワ	4F4		倭	和	話	歪	賄	脇	惑	枠	鷲	亙	亘	鰐	詫	藁	蕨
	4F5	椀	湾	碗	腕												
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

付．3 エラーコード一覧表

注意：エラーメッセージの欄で，フルセンテンスで示されているものは，N_{88} Disk-BASIC または N-BASIC のメッセージであり，カッコ内に示されている省略形のメッセージは N_{88}-BASIC のものです．エラーコードの欄は上段が N-BASIC，下段が N_{88}-BASIC のコードを表わしています．"—"で示されているのは，存在しないエラーです．

エラーメッセージ	コード	意味
Bad allocation table	N 64 N88 69	ディスケット内部のFATが壊れている．
Bad drive number	N 65 N88 70	ドライブ指定が誤っている．システムにつながっていないドライブを指定した．
Bad File Data	N 25 —	ファイル上にあるデータの形式がまちがっている．
Bad file name (? NM Error)	N 62 N88 56	ファイル名の指定がまちがっている．
Bad file number (? BN Error)	N 52 N88 52	オープンしていないファイルや起動時に指定していないファイルを参照した．
Bad file mode	N 54 —	シーケンシャルでオープンしたファイルに対してランダムアクセスをしようとした．またはその逆．
Bad record number	N 63 —	ランダムアクセスファイルで許容されるレコードナンバーから外れている．
Bad track/ sector	N 66 N88 71	トラック，セクタ番号の指定が誤っている．（DSKO $, DSKI $)
Can't continue (? CN Error)	N 17 N88 17	CONT命令によって実行を再開することができない（ポインタの値が壊されている）．
Communications Buffer Overflow	N 27 —	周辺機器との入出力のためのバッファがオーバーフローした．（N88-BASICではLine buffer overflow）
Deleted record	N 67 N88 72	消去済みのレコードをアクセスしようとした．
Direct Statment in file (? DS Error)	N 63 N88 57	アスキー形式のファイルを読み込む際にダイレクトステートメントが存在した．
Disk already mounted	N 59 N88 67	mountされているディスクに対してmountを行った．
Disk BASIC Feature	N 26 —	ディスクが接続されていないとき，ディスクBASICの命令を実行した．

エラーメッセージ	コード	意　　味
Disk full	N 60 N88 68	ディスク上に書き込むスペースがないのに書き込もうとした．
Disk not mounted	N 56 N88 63	mount されていないディスクに対してアクセスした．
Disk I/O Error	N 57 N88 64	ディスクとの入出力中にエラーが発生した．致命的なエラーであり回復させることはできない．
Disk offline	N 73 N88 62	入出力の可能な状態でないディスクをアクセスしようとした．
Division by Zero (? /0 Error)	N 11 N88 11	0による割り算が実行されて，その結果の値がオーバーフローした．
Duplicate Definition (? DD Error)	— N88 10	配列またはユーザー関数を二重定義しようとした．（N-BASIC では，Redimensioned array）
duplicate label (? DU Error)	— N88 31	同じラベル名が二つ以上存在している．
Feature not available (? NA Error)	— N88 33	利用不可能な機能を指定した．
FIELD overflow (? FO Error)	N 50 N88 50	FIELD 文において256バイト以上の大きさの領域を指定した．
File already exists	N 58 N88 65	NAME 文によって変更しようとしたファイル名が既に存在している。
File already open (? AO Error)	N 55 N88 54	既にオープンされているファイルに対して OPEN，KILL などを実行した．
File not found (?FF Error)	N 53 N88 53	LOAD，SAVE，KILL などで現在のディスクに存在しないファイルを指定した．
File not Open (? CF Error)	N 71 N88 60	オープンされていないファイルを参照しようとした．
File write protected	N 72 N88 61	書き込み禁止属性が付けられているファイルに書き込もうとした．
For without NEXT (? FN Error)	— N88 26	FOR～NEXT が正しく対応していない（FOR が多い）．
Illegal direct (? ID Error)	N 12 N88 12	ダイレクトモードで使用できない文を実行しようとした．

エラーメッセージ	コード	意　味
Illegal function call (? FC Error)	N 5 N88 5	ステートメントおよび関数において，機能の呼び方が誤っている．
Input past end (? EF Error)	N 61 N88 55	ファイル中のすべてのデータを読み尽くした後に，さらに入力文が実行された．
Internal error (?IE Error)	N 51 N88 51	BASIC 内部にエラーが生じた．
Line buffer overflow (?BO Error)	N 23 N88 23	1行で入力できる文字の範囲を超えて入力が行われた．
Missing operand (? MO Error)	N 22 N88 22	ステートメント中必要なパラメータが指定されていない．
NEXT without FOR (? NF Error)	N 1 N88 1	FOR～NEXT が正しく対応していない． (NEXT が多い)
No RESUME (?NR Error)	N 19 N88 19	エラー処理ルーチンの中に RESUME 文がなく，プログラムの実行が継続できない．
Out of data (? OD Error)	N 4 N88 4	読むべきデータがないのに READ 文が実行された．
Out of memory (? OM Error)	N 7 N88 7	メモリ容量が足りなくなった（プログラムが大きすぎる，またはスタックを消費し尽した）．
Out of string space (? OS Error)	N 14 N88 14	文字列を格納するメモリ領域がなくなった．
Overflow (? OV Error)	N 6 N88 6	演算結果や入力された数値が，許される範囲を超えた．
Port not initialized	N 28 —	インターフェイス用の LSI の機能設定がなされていない．
Position not on Screen	N 24 —	指定したカーソル位置などが，画面の範囲外になっている．
Redimensioned array	N 10 —	配列またはユーザー関数を二重定義しようとした（N88 -BASIC では Dupulicate Definition）．
Rename across disks	N 68 N88 73	NAME 文において，異なるドライブ間でリネームしようとした．
RESUME without error (? RW Error)	N 20 N88 20	エラー処理ルーチンに制御を移さなかったのに RESUME 文がある．

エラーメッセージ	コード	意　　　味
RETURN without GOSUB (? RG Error)	Ⓝ 3 N88 3	GOSUB～RETURN が正しく対応していない（RETURN が多い）.
Sequential after PUT (? SP Error)	Ⓝ 69 N88 58	PUT 文実行後，シーケンシャルファイルをアクセスしようとした.
Sequential I/O Only (?SQ Error)	Ⓝ 70 N88 59	シーケンシャル入出力以外は行ってはならない.
String formula too complex (? ST Error)	Ⓝ 16 N88 16	文字式が複雑すぎる. （カッコのネスティングが深すぎるなど）
String too long (? LS Error)	Ⓝ 15 N88 15	一つの文字変数内の文字数が 255 文字を超えている.
Subscript out of range (? BS Error) (Bad Subscript)	Ⓝ 9 N88 9	配列変数の添字の値が規定の範囲から外れている.
Syntax error (? SN Error)	Ⓝ 2 N88 2	文の記述が文法どおりになっていない.
Tape read error (? TP Error)	Ⓝ 29 N88 27	テープからの読み込み時にエラーが発生した.
Too many files	Ⓝ 67 —	ファイルの数が許容される範囲（255個）を超えた.
Type mismatch (? TM Error)	Ⓝ 13 N88 13	式の左辺・右辺，関数の引数などにおいて変数の型が一致していない.
undefined label (? UN Error)	— N88 32	定義されていないラベル名で参照した.
Undefined line number (? UL Error)	Ⓝ 8 N88 8	飛び先として必要とされる行番号（GOTO, GOSUB などの飛び先）が存在しない.
Undefined user function (? UF Error)	Ⓝ 18 N88 18	DEFFN 文によって定義されていないユーザー関数を引用しようとした.
Unprintable error (? UE Error)	Ⓝ 21 N88 21	メッセージの定義されていないエラー.
WEND without WHILE (? WE Error)	— N88 30	WHILE～WEND が正しく対応していない. （WEND が多い）.
WHILE without WEND (? WH Error)	— N88 29	WHILE～WEND が正しく対応していない. （WHILE が多い）.

〔索　　　引〕

ア　行

カ　行

サ　行

タ　行

〈コマンド・命令・関数〉

〈参考にした図書〉

- PC-8801mkII　BASICリファレンスマニュアル（NEC）
- PC-8801mkII　ユーザーズマニュアル（NEC）
- PC-8800　BASIC入門編（電子開発学園）
- PC-8800　BASIC初級編（電子開発学園）
- PC-8800　BASIC中級編（電子開発学園）
- PC-8001mkII　BASIC　上巻（電子開発学園）
- PC-8001mkII　BASIC　下巻（電子開発学園）

監修者略歴

工学博士　松　尾　三　郎（まつお さぶろう）

〔略　歴〕

昭和13年3月	京都帝国大学工学部電気工学科卒業
昭和20年9月	逓信省電波局勤務，電波課調査係長
昭和22年3月	京都帝国大学工学部講師（～30年3月）
昭和24年11月	電気通信省電気通信研究所方式実用化部電波課長
昭和28年3月	日本電信電話公社電気通信研究所方式部無線課長（～29年4月）
昭和29年4月	ニッポン放送技術局次長（～34年3月）
昭和32年4月	武蔵工業大学教授（～39年3月）
昭和32年5月	日本電波塔株式会社取締役技術部長（～35年4月）
昭和34年4月	日本技術開発株式会社専務取締役（～39年9月）
昭和37年8月	日本ビジネスオートメーション株式会社取締役副社長（～39年9月）
昭和39年4月	北海道ビジネスオートメーション株式会社取締役社長（～44年6月）
昭和40年5月	日本電子開発株式会社取締役社長
昭和45年4月	電子開発学園学園長
昭和51年4月	郵政省電波技術審議会委員
現　在	株式会社イーディシー取締役社長
	電子開発学園学園長
	日本電子開発株式会社取締役社長
	ソフトウェアコンサルタント株式会社取締役社長
	郵政省電波技術審議会委員

PC-8801mkIIの使い方　下巻　　　**定価2,500円**

昭和60年1月15日　初版第1刷発行
昭和60年2月20日　第2刷発行

監　修／松　尾　三　郎
編　集／電子開発学園 出版局
発行者／松　尾　三　郎
発行所／㈱イーディシー
電子開発学園 出版局
〒164 東京都中野区中野5丁目62番1号（ＥＤＣビル）
電話 (03) 319－7101
振替　東京2－54892
印刷・製本所／(株)カントー
電話(03)238-6011

ISBN4-88647-024-6

PC-8801mkII(マーク)の使い方 下巻　（定価 2,500円）

PC-8801mkII（マーク）の使い方 下巻

ISBN4-88647-024-6 C3055 ¥2500E

定価 2,500円